Center for eBusiness@MIT
http://ebusiness.mit.edu

A research and education initiative at the MIT Sloan School of Management

A Handbook-Based Methodology for Redesigning Business Processes

Paper 221

January 2006

Mark Klein
Claudio Petti

For more information,
please visit our website at **http://ebusiness.mit.edu**
or contact the Center directly at **ebusiness@mit.edu or 617-253-7054**

MITSloan

A Handbook-Based Methodology for Redesigning Business Processes

Mark Klein
Sloan School of Management
Massachusetts Institute of Technology
Cambridge MA USA
m_klein@mit.edu

Claudio Petti
e-Business Management School
ISUFI - Università di Lecce
Lecce, Italy
claudio.petti@ebms.unile.it

Abstract

This paper presents a structured methodology, based on the use of a Handbook of process models, for redesigning business processes. The methodology is illustrated using examples from the agri-food supply chain domain. We discuss the strengths and weaknesses of this approach, and identify avenues for further work.

The Challenge: Enabling Innovation in Process Reengineering

Organizations nowadays are under increasing pressure to adapt their business processes to relentless technological, political, organizational, and other changes (Davenport and Perez-Guardado 1999). Under such conditions, being able to rapidly generate good new ideas about how to meet these challenges becomes a critical skill.

A body of process innovation techniques known collectively as Business Process Re-engineering (BPR) has emerged to address this challenge (Armistead & Rowland 1996; Chen 1999; Davenport and Short 1990; Hammer 1990; Grover et al. 1995; Hammer & Champy 1993; Kettinger et al. 1997b; Kubeck 1995, 1997; Nissen 1998, 1999; Pandya and Nelis 1998). Despite the widespread use of these tools, however, many process innovation initiatives fall short of delivering the hoped-for results. While they typically aim for revolutionary change, they often result in only incremental improvements (Stoddard and Jarvenpaa 1995).

We can understand why this is so by considering the nature of current BPR techniques. A comprehensive survey of these practices (Kettinger et al 1997b) identified the following categories of tools relevant to the redesign stage of process innovation:

- IDEF modeling
- Data modeling, including data flow diagramming, flow charting, case-based information engineering tools
- Process simulation
- Creativity techniques, including brainstorming, out-of-the-box thinking, nominal group, visioning, etc.

Most of these tools - including IDEF, data modeling, and process simulation - are oriented towards modeling or analyzing *as-is* business processes rather than defining new ones. The design of *to-be* processes is supported only by generic creativity techniques such as brainstorming and visioning. These techniques are useful for producing novel ideas, but since they rely only on what happens to be on the minds of the participants, they are unlikely to support systematic exploration of a full range of alternatives (Lee and Pentland 2000;

Pentland 1995). All this leads to a tendency to lavish effort on documenting and refining the existing business processes (what Hammer and Champy call “analysis paralysis” (Hammer and Champy 1993)) rather than defining radical new alternatives to these processes (Lee & Pentland 2000) (Klein et al 2003). The re-designed business processes take too long to develop, tend to lag behind changes in the environment, and often represent minor variations of processes the designers are already familiar with and have spent so much time painstakingly documenting.

This paper presents a BPR methodology designed to address these weaknesses in existing techniques. The methodology is based on acquiring an abstract model of just the *core* activities and dependencies in the existing process, and then engaging in a structured and systematic exploration of process alternatives, utilizing for inspiration a large repository of best-practice business processes. The paper will present the methodology, drawing on illustrative case examples from our work with agri-food supply chains, consider the lessons we have learned so far, and suggest directions for future work.

Background: The Process Handbook

Our BPR methodology is based upon the Process Handbook, an electronic repository of best-practice business processes. The result of over a decade of development by over 40 researchers and practitioners centered around the MIT Center for Coordination Science, the Handbook includes a database of over 5000 business processes in addition to software tools for viewing, searching, and editing the database contents (Malone et al. 1999) (Malone et al. 2003). The Handbook utilizes several key concepts:

Process Specialization: Practically all process representation techniques (including ours) use the notion of decomposition, i.e., that a process can be broken down (or "decomposed") into sub-activities. Our representation includes in addition to this the concept of *specialization*. While a sub-activity represents a *part* of a process, a specialization represents a type or *way of doing* the process (Taivalsaari 1996; van der Alst & Basten 1999; Wyner & Lee 2000). Using this concept, processes can be arranged hierarchically into a taxonomy, with very generic processes at one extreme and increasingly specialized processes at the other. As with other taxonomies, specialized entities automatically inherit properties from their more generic "parents", except where they explicitly add, delete or change a property.

Figure 1 illustrates this approach. Here, the generic process "Sell product" is decomposed into sub-activities like "Identify potential customers" and "Inform potential customers”. The generic process is also specialized into more focused process like "Sell by mail order" and "Sell in retail store". These specialized processes inherit, by default, the sub-activities and other characteristics of their "parent" process (AKA *generalization*):

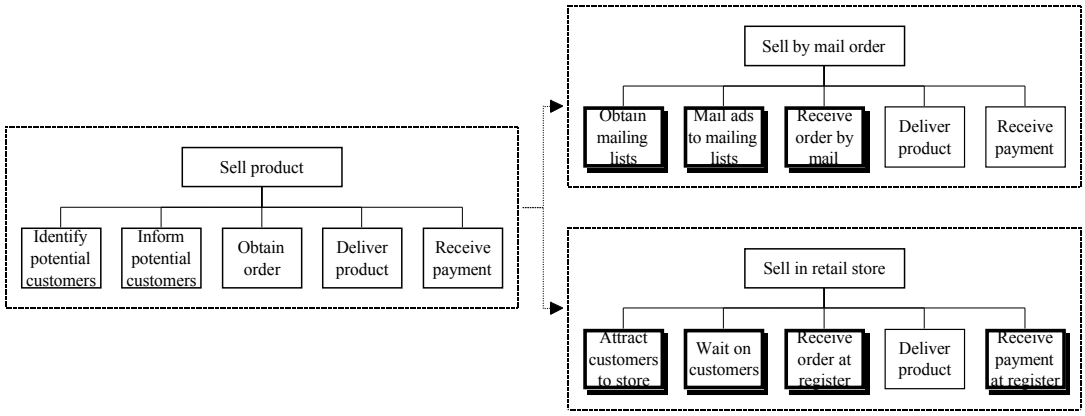

Figure 1. An example of inheritance in the specialization hierarchy.
Adapted from (Malone et al. 1999).

Specialized processes can also add to or change the sub-activities they inherit. For instance, in "Sell by mail order", the sub-activities of "deliver a product" and "receive payment" are inherited without modification, but "Identify potential customers" is replaced by the more specialized activity "Obtain mailing lists."

In addition to processes like “sell by mail order” that can be viewed as general templates, the specialization hierarchy also includes “case studies” documenting creative solutions developed by organizations in response to particular process challenges. We have captured roughly 400 such case studies to date, ranging from “Hire human resources in advance of need {L-S Electro-Galvanizing}” to “Make using vendor assembly {VW Brazil}”. Capturing such creative solutions in the repository enables their wider adoption by others.

We have found it useful to combine specializations into what we call "bundles". Each bundle represents an orthogonal dimension along which the specializations of a process can vary. Generally speaking, the specializations under a bundle represent alternative answers to the question posed in the bundle. One can thus speak of “how” bundles (that gather different techniques for *how* the activity is performed), “who” bundles (that represent different alternatives for *who* performs an activity), “what” bundles (that represent different alternatives for *what* resource is manipulated by the activity), and so on. Figure 2 shows several examples of bundles in the specialization hierarchy for the “Sell” process, including "Sell how?" (which collects alternatives for how the sale is made), and “Sell What?” (which collects alternatives for what is sold):

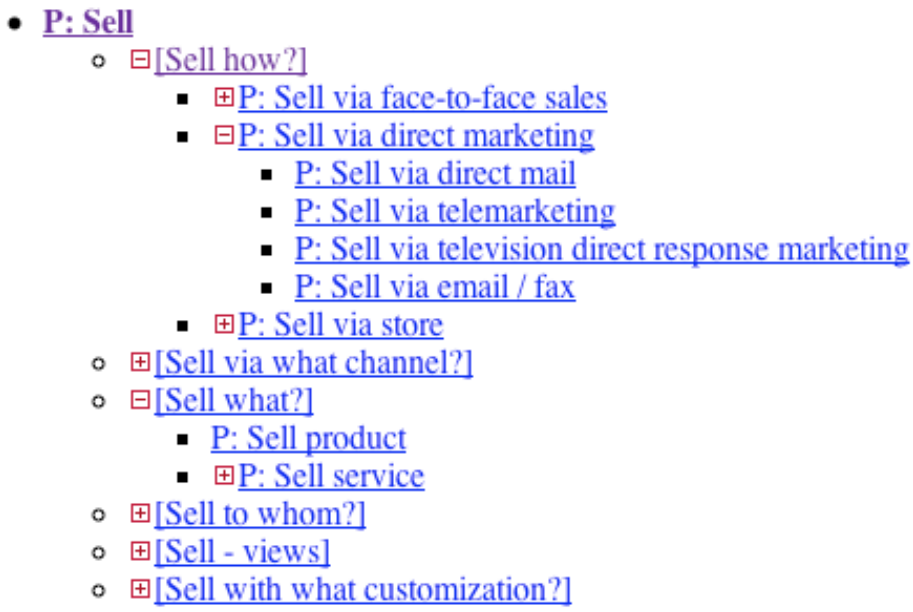

Figure 2. An example of bundles in the specialization hierarchy.

Bundles can have associated tradeoff tables that capture the relative pros and cons of the different specializations in terms of their ratings on various criteria. Figure 3, for example, shows a tradeoff table for the specializations in the "Sell How?" bundle; specializations are the rows, criteria are the columns, and the cell contents are the values for each criterion and specialization:

Alternative	Suggested products	Quality of service	Time to sell	Cost of selling
P: Sell via face-to-face sales	high margin, tailored	high	medium	high
P: Sell via direct marketing	specialty items	low	long	low
P: Sell via store	low margin commodities	medium	medium	medium

Figure 3. An example of a tradeoff table.

The power of the specialization hierarchy comes from the fact that processes with similar functions are colocated, irregardless of the context (e.g. which industry) the process originally comes from. This enables the cross-disciplinary fertilization of ideas and the leveraging of experiences across different industries.

Dependencies and Coordination Mechanisms: A second key concept is the notion that coordination can be viewed as the process of managing the resources (including documents, physical resoures such as fuel, signals, and so on) that are shared, in some way, by the sub-activities in a process. We call these resource relationships *dependencies,* and distinguish dependencies into three basic types: flow, sharing and fit (Malone and Crowston 1994) (Crowston 1991) (Figure 4). *Flow* dependencies arise whenever one activity produces a resource that is consumed by another activity. Every flow dependency has three components: *timing* (ensuring the flow occurs at the right time), *accessibility* (ensuring the flow goes to the

right place) and *usability* (making sure the right resource is transfered). *Sharing* dependencies occur whenever multiple activities all use the same scarce resource (e.g. when two people need to use the same machine). *Fit* dependencies arise when multiple activities collectively produce the components of a single resource (e.g. when several designers create subcomponents for a single system).

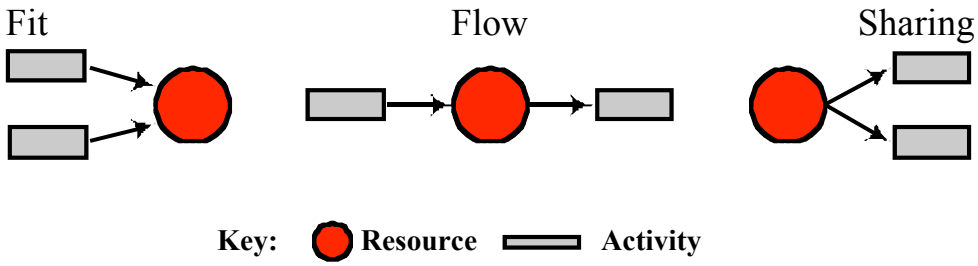

Figure 4. Three basic types of dependencies among activities.
Adapted from (Malone et al. 1999)

Figure 5 gives an example of these dependencies as they occur in the "Sell" process:

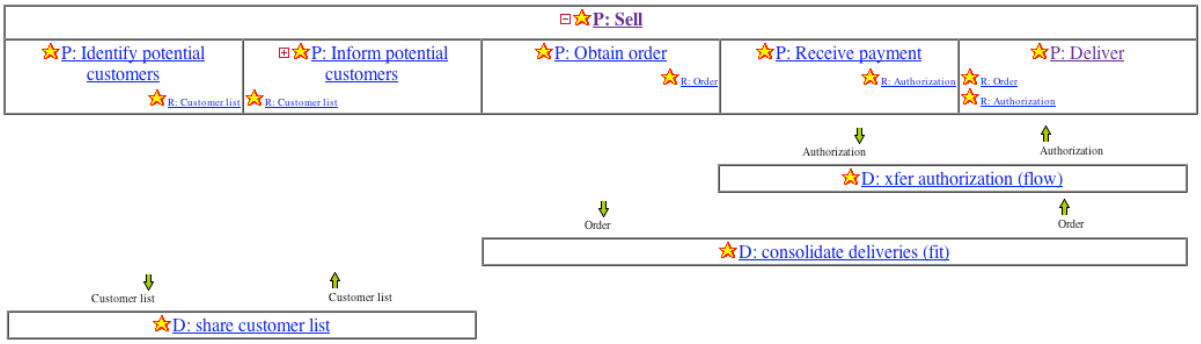

Figure 5. Dependencies in the process "Sell".

One dependency concerns the "customer list" resource. Assuming multiple sales agents, we need to manage the sharing of customer lists amongst these agents. A second dependency concerns flow: the right delivery authorizations need to be transfered to the right delivery agents at the right time. Finally, there is a fit dependency between delivery authorizations: if there are multiple orders from one customer, it makes sense to try to consolidate these multiple orders for delivery purposes.

The resource relationships represented by such dependencies are managed by processes we call *coordination mechanisms*. As Table 1 illustrates, each dependency type has its own set of mechanisms potentially relevant for managing it (Malone et al, 1999):

Dependency	Examples of coordination mechanisms for managing dependency
Flow: Timing	Transfer resource periodically Transfer resource on demand Transfer resource as it is generated
Flow:Access	Ship to consumer Make at point of use
Flow:Usability	Producer follows standards Consumer filters out sub-standard resources
Sharing	First come/first served Market-like bidding
Fit	Predefined subsystem interfaces Concurrent engineering design teams

Table 1. Examples of dependencies and associated coordination mechanisms.

By consulting Table 1 we can see, for example, that the sharing dependency in the "Sell" process can be managed by allocating customer names to sales agents on a first-come, first-served basis, or by sales agents bidding for customer names using some kind of market.

Exceptions and Handlers: The third key concept underlying the Handbook captures how processes deal with potential failures (Klein et al. 2003b). All processes in the Handbook repository are linked to the possible ways (*exceptions*) that the process may fail to achieve its goals. These exceptions, like all other process attributes, are inherited down the process specialization hierarchy. Consider, for example, the exceptions associated with "Inform potential customers", a sub-activity of "Sell" (Figure 6):

Goal	Exception
Unspecified	
G: process achieves optimal outcome	
G: process avoids negative side effects	E: unwanted solicitation
G: process uses minimal resources	
G: process terminates in finite time	E: Performing agent dies

Figure 6: Exceptions for the process "Inform Potential Customers"

Every process in the Handbook inherits the exception "Performing agent dies" (which violates the goal "process terminates in finite time"), while the "unwanted solicitation" exception (which violates the goal "process avoids unwanted side effects") is specific to the "Inform potential customer" process and its specializations.

Exceptions are linked, in turn, to the processes (called *handlers*) potentially relevant to managing (i.e. anticipating and avoiding, or detecting and resolving) them. The exception "performing agent dies", for example, is linked to the following handlers (Figure 7):

Handler
ANTICIPATED-BY *P: Track MTBF*
AVOIDED-BY *P: Filter out low-reliability resources*
DETECTED-BY *P: Monitor resource for failures*
RESOLVED-BY *P: Replace with other resource*
RESOLVED-BY P: Resurrect agent

Figure 7. Handlers for the exception "performing agent dies".

These links show that one can *anticipate* agent failure by tracking the MTBF (mean time between failure) for that kind of agent, *avoid* failure by filtering out agents that are known to be unreliable, *resolve* the failure by replacing the failed agent, and so on.

The Handbook Business Process Redesign Methodology

The Handbook approach to BPR takes advantage of a process repository, organized using the concepts described above, to enable the systematic exploration of potential re-designs for a given process. The methodology described herein represents an integration and refinement of previously distinct techniques for designing the normative (Malone et all 1999) (Klein et al 2003) and exception-handling (Klein et al 2003b) elements of a business process. Our approach begins by creating a "stripped-down" model of the as-is process, one that captures only the core activities and key dependencies. One then uses the Handbook repository as a source of ideas concerning how the activities can be realized, the dependencies managed, and the possible exceptions handled. We will describe this procedure in more detail below, using a simple example drawn from the agri-food supply chain domain.

Step 1 - Identifying the Process 'Deep Structure'

The first step in re-designing a process is to identify a good initial abstraction of that process, what we can call the "deep structure". This involves identifying the core activities (i.e. those activities which are core to the definition of the business) and key dependencies (i.e. the resource relationships that must appear between the core activities). Our goal is to capture the essence of the as-is process, rather than become enmeshed in capturing details that we will probably want to radically re-design anyway.

A first cut at a deep structure for the agri-food supply chain, for example, might look like the following (Figure 8):

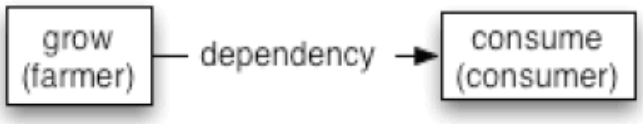

Figure 8. A deep structure for the agri-food supply chain.

There are two core activities: grow food (performed by farmers), and consume food (performed by consumers). There is also one key dependency, that manages the movement of

food from growers to consumers. This dependency includes several components, including sharing (e.g. determining which consumers get a farmer's goods), flow (e.g. determining when the goods are transferred) and even fit (e.g. making sure that the each consumer gets goods that fit together to create the meals they want).

We may choose to add a core activity called "process" to the agri-food supply chain, since almost all food products are processed in some way before being consumed (Figure 9):

Figure 9. The food supply chain with a "process food" step.

We may also add, finally, a core activity called "distribute", representing the activity of entities that intermediate between food processors and consumers (Figure 10):

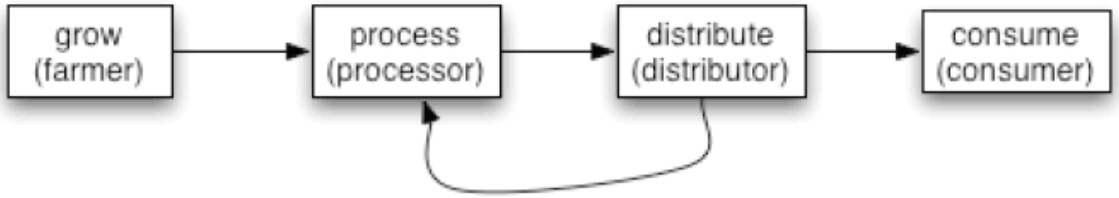

Figure 10. The food supply chain with a "distribute" step.

A repository of business process models like the Handbook can prove valuable during the capture of the as-is deep structure, because it provides a collection of pre-defined process "building blocks". It is often quicker to hook together existing components rather than define them from scratch. Building blocks can also foster greater completeness by virtue of including elements (e.g. sub-activities or exceptions) that we might otherwise forget. The Handbook repository's model for "distribute", for example, is the following (Figure 11):

⊟P: Distribute											
⊟P: Buy							⊟P: Sell				
P: Identify own needs	P: Identify potential sources	P: Select supplier	P: Place order	P: Receive	P: Pay	⊞P: Manage suppliers	P: Identify potential customers	⊞P: Inform potential customers	P: Obtain order	P: Receive payment	P: Deliver

Figure 11. A decomposition for the "Distribute" process.

If we use this process model as a building block, the agri-food supply chain deep structure we build will automatically include such sub-activities as "identify own needs", "identify potential sources", and so on. But most importantly, as we shall see below, using the existing Handbook building blocks allows us to tap into the Handbook's rich collection of best practice models when seeking alternatives to how our business process is currently designed.

Since the Handbook repository is fairly large (over 5000 processes, and growing), good search tools are needed to make it easy to find the right building blocks. The Handbook software includes several such tools, including a browser for the specialization hierarchy, a collection of folders (called "Guided Tours") that collect processes relating to a given theme (e.g. supply chains). a keyword search capability, as well as a sophisticated "graphical search" engine known as PQL (the Process Query Language) that allows one to search for entities that have a given set of inter-relationships (Klein et al. 2004). The following PQL query, for example, allows you to search for all "sell" processes that have a sub-activity that uses the Internet:

```
((entity ?top isa process)
 (attribute "Name" of ?top includes "sell")
 (relation ?top has-specialization ?p *)
 (relation ?p has-part ?part *)
 (relation ?part has-enabler ?enabler)
 (relation ?gen has-specialization ?enabler *)
 (attribute "Name" of ?gen includes "Internet"))
```

PQL has proven to be a powerful search facility but, as with any formal query language, the queries can be verbose, and significant technical expertise is needed to define them correctly. The Textual Query Language (TQL) was created to address this limitation by providing a simple, compact, English-like syntax suitable for expressing a wide range of PQL queries. The user enters queries in TQL, which is then automatically translated into PQL and enacted by the PQL interpreter. The query above, for example, can be expressed in TQL as follows:

```
process isa "sell"
 has-part
  has-enabler isa "Internet"
```

Step 2 – Specializing core activities

Once the deep structure has been captured, we can start exploring different ways of refining this abstract model into a fully-specified business process. The first step is to replace the abstract core activities in the deep structure model with specific ones. We can replace the "distribute" activity above, for example, with the more specialized "distribute using grocery store" process.

The ideas for how we specialize the core activities can, of course, be generated by the BPR participants based solely on their own experience, augmented perhaps by such generic 'creativity' tools as brainstorming and the like. The shortcoming of this approach, however, is that the quality of the re-designed process is limited to whatever alternatives the participants happened to think of during this particular engagement. In the absence of a systematic procedure for enumerating potentially useful alternatives, superior designs may be missed. This represents the key contribution of the Handbook's process repository. Building block processes in the Handbook have, as their specializations, collections of "best practice" ideas, gathered from many sources and industries. Rather than having to generate process alternatives based on our individual experiences, we can simply *select* from the process ideas in the Handbook.

Imagine, for example, that we are defining how the activity "identify potential sources" (a sub-activity of "distribute") is realized. "Identify potential sources" heads the following branch in the Handbook's specialization hierarchy (Figure 12):

Figure 12. The "identify potential sources" branch in the specialization hierarchy.

To specify how our process will identify potential sources, we need only select one of the specializations from this branch. We can use the tradeoff tables to help us decide which specialization best suits our current needs. We can also compare the different specializations in terms of the exceptions they are prone to, and how easy these exceptions are to handle.

We can streamline the specification procedure by allowing process designers to select specializations for all the deep structure's sub-activities in parallel. The tool we developed for this purpose, which we call the "part recombinator", allows you to refine a deep structure model in much the same way you select a meal at a restaurant: you select one option for the first course (first sub-activity), another option for the second course (second sub-activity), and so on until you have made one selection for every stage of the meal (i.e. for every activity in the process). Figure 13 below shows an example of the part recombinator applied to the agri-food supply chain deep structure. Each column shows the specialization hierarchy for the one sub-activity (the one whose name is given in bold at the top of the column). Checked-off entries represent the specializations selected for the sub-activity in that column:

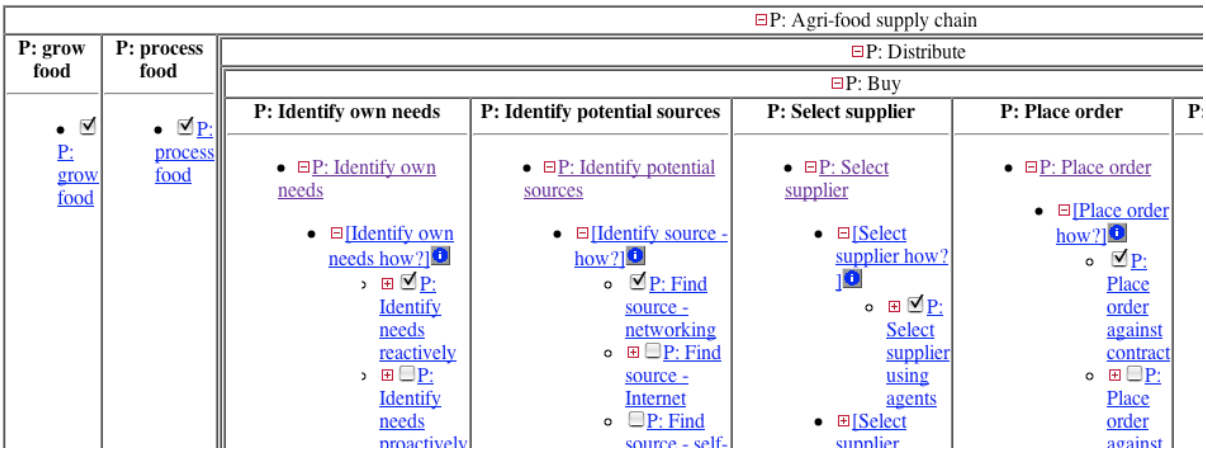

Figure 13. The part recombinator applied to the agri-food supply chain deep model. Only a portion of the recombinator display is shown in this figure.

In this case, "identify own needs" was specialized into the process "identify needs reactively", "identify potential sources" was specialized into "find source via networking", and so on. The process model defined as a result of these selections is given below (Figure 14):

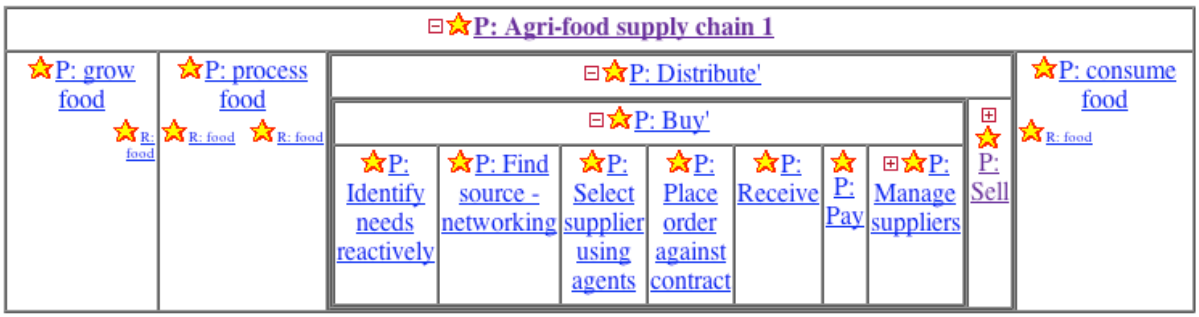

Figure 14. The process model defined by the selections made in figure 13 above.

It should be noted that this kind of "design by selection" is not a rote mechanical process. The specializations in the Handbook represent ideas only, intended to trigger a creative process wherein the process designers consider whether and how these ideas, or some combination/variant thereof, might be applied in their particular business context. The repository thus serves to enhance creativity, rather than replace it.

Step 3 – Specifying coordination mechanisms

In addition to specifying how the core activities in the deep structure model are realized, we also need to specify how the dependencies between these activities are managed. Every dependency type (flow vs fit vs sharing) has, in the Handbook repository, a corresponding branch of the specialization hierarchy that captures the processes (AKA "coordination mechanisms") that can manage that dependency. There is a "manage fit" branch of processes suited for managing fit dependencies, a "manage sharing" branch for managing share dependencies, and so on. To specify the coordination mechanism for a given dependency,

then, all we need do is select the coordination mechanism we want from the corresponding branch of the specialization hierarchy.

Let us consider, for example, the dependency between “grow food” and “process food” in the agri-food supply chain (Figure 10). This dependency includes a “share” component, since we need to decide how the goods produced by each farmer are shared among the processors that transform these goods. We can specify how this dependency is managed by selecting a coordination mechanism from the “manage sharing” branch of the repository (Figure 15):

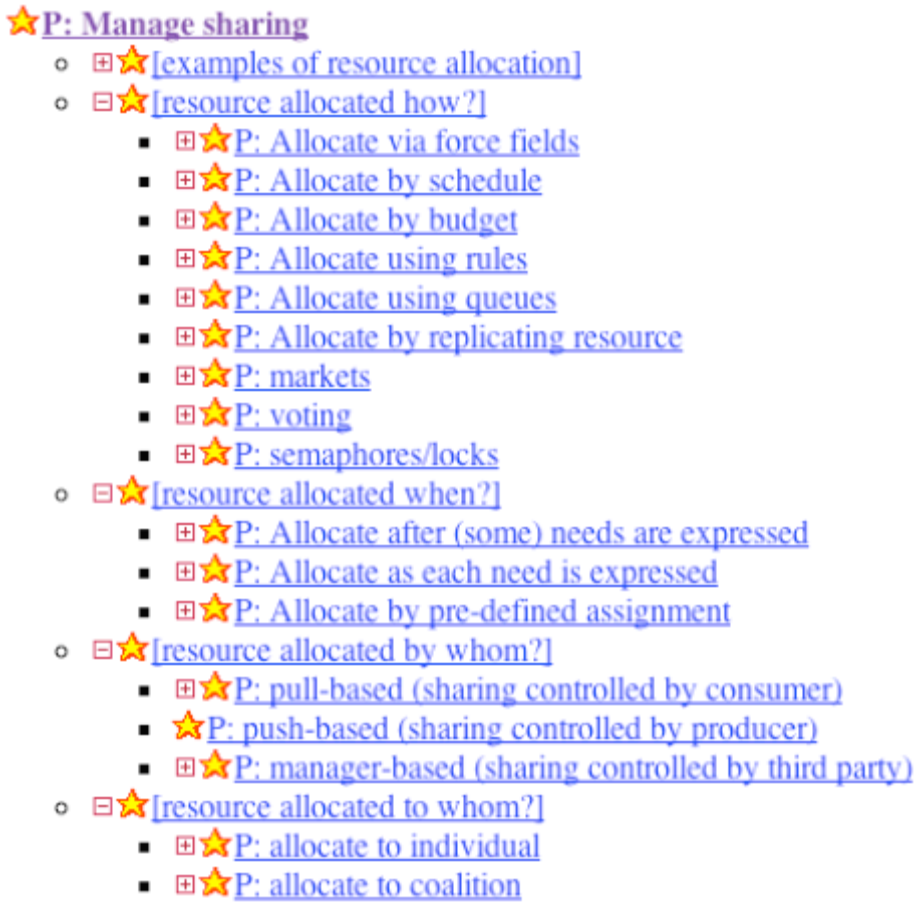

Figure 15. The top level of the “manage sharing” branch of the specialization hierarchy.

The structure of this branch reveals an important aspect of using the Handbook for BPR. The Handbook makes use of a powerful concept we call ‘bundle recombination’ to enable the generation of process alternatives. Recall that every bundle (e.g. “resource allocated how?”, “resource allocated when?” and so on) represents an orthogonal dimension over which one can classify specializations of a given process. The bundles and their specializations thus specify a multi-dimensional design space of possible processes. We can therefore generate new process alternatives by making one (or possibly more than one) choice from each bundle. We can, for example, consider a sharing mechanism based on budgets (how? bundle) where the resources are allocated after needs are expressed (when?) by a manager (by who?) to individual consumers (to whom?). Not every combination of selections will represent a superior or even workable process alternative, but the bundle recombination approach does enable a systematic approach to exploring the design space and may lead us to consider alternatives that would otherwise have been overlooked.

Step 4 – Specifying exception handlers

The final stage of the Handbook BPR methodology involves specifying how exceptions should be handled in the process we are designing. We first need to identify, for every activity in the process (core activities as well as coordination mechanisms), which exceptions are of concern. Recall that all processes in the Handbook repository are linked to the exceptions that may affect them, and these exceptions are linked in turn to potentially applicable handlers. If we used processes from the Handbook repository as building blocks for the deep structure model, we simply need identify, from a pre-enumerated list of exceptions, which ones are important in our particular context. Having done that, we can then select, from a pre-enumerated list of handlers, which ones we want to use for the selected exceptions. We have developed, for this purpose, a tool we call the "exception recombinator". Figure 16 (below) shows an example of using the exception recombinator for the "agent death" exception:

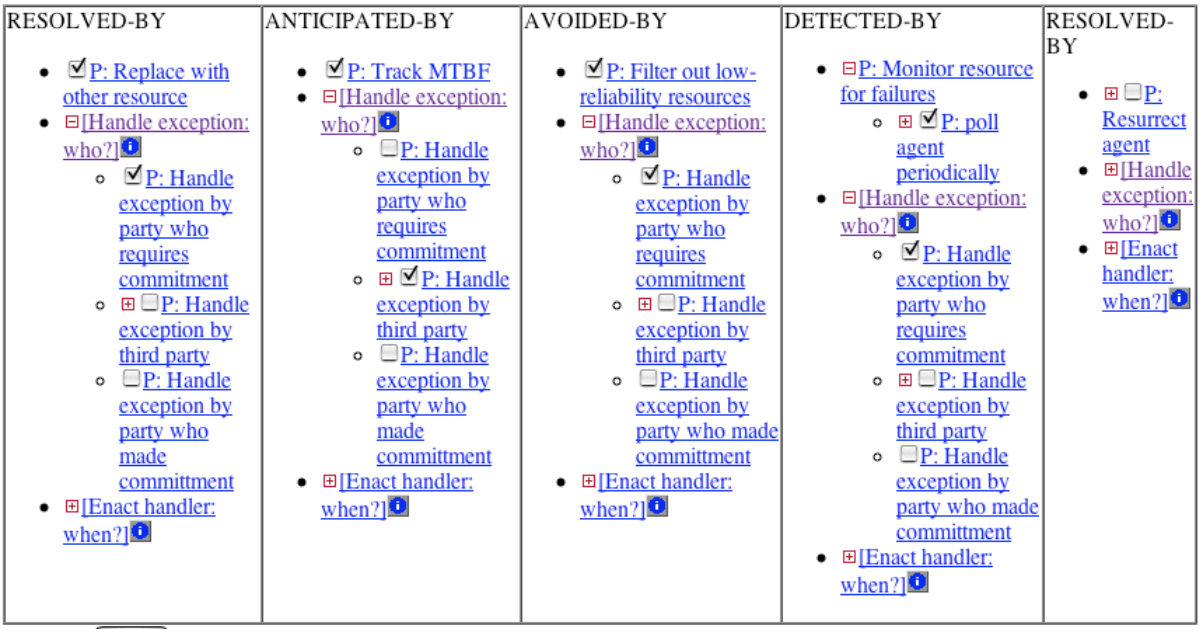

Figure 16. Using the exception recombinator to specify a handler for "agent death".

Each column of the exception recombinator lists one of the handlers relevant to a given exception, as well as all of its specializations. You can select which handlers you want to use simply by putting a check next to their name. The selections in figure 14, for example, result in the creation of the following exception handler process (Figure 17):

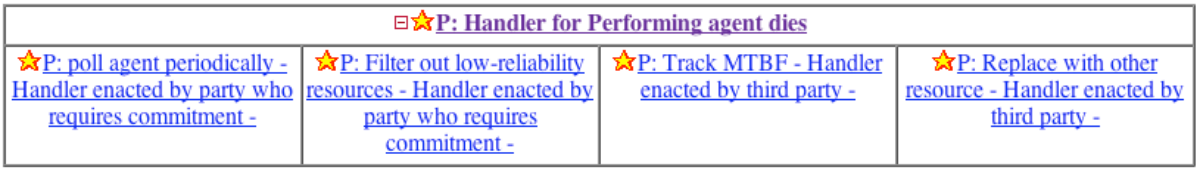

Figure 17. The exception handling process specified by the selections in figure 16.

Since exception handler processes that may face their own exceptions, a thorough BPR engagement will consider how the handler's potential exceptions can themselves be handled.

Summary

The Handbook BPR methodology can be summarized as follows (Table 2).

BPR Step	Process Analyst's Role	Domain Expert's Role
Identify "deep" structure	Help domain expert abstract away surface structure features.	Describe core processes and key dependencies. Ensure model captures all relevant activities, resources, and actors.
Enter deep structure model into Handbook repository	Enter the model into the database, using the closest matches in the taxonomy as building blocks.	Validate, as needed, whether the appropriate building blocks were used.
Re-design processes by recombination	Use the bundle, part and exception recombinators to suggest potentially relevant new process ideas.	Critique, prune, and refine process ideas. Suggest new options inspired by those generated from the database.

Table 2: Process Handbook BPR Methodology Steps and Roles

We can distinguish two key roles: process analyst and domain expert. The process analyst is fluent with the re-design methodology and familiar enough with the Handbook repository to help make sure the deep structure model is built from the right building blocks. The domain expert, by contrast, is knowledgeable about the business process being re-designed, and is able to identify especially promising re-design alternatives from among the ones generated using the BPR methodology.

Progress to Date

The MIT Handbook project has been active for over a decade, and in that time the Handbook has been used and refined by an internationally-distributed group of researchers, students and practitioners. The Handbook BPR methodology has been applied to hiring processes at a major financial services firm (Klein et al. 2003) as well as to logistics processes at Gillette and Nestle. It is currently being applied, as part of an ongoing collaboration between MIT and the ISUFI School of e-Business Management, to helping small and medium-sized firms design and implement their transition towards e-business.

The Handbook software includes both Windows and Web-based versions. Both versions support viewing and editing the database. The web version includes, in addition, the PQL and TQL query engines, and is available on-line at http://franc2.mit.edu:8000/pql/. General information about the Process Handbook project is available at http://ccs.mit.edu/ph/.

The Handbook process repository now includes over 5000 process models covering a wide range of business functions. We have focused the bulk of our efforts on capturing process knowledge related to coordination. Virtually all organizations face the challenge of enabling

effective coordination, i.e. of ensuring that the right individuals do the right tasks at the right time using the right resources. Coordination challenges for organizations of all kinds have grown increasingly severe, driven by ubiquitous telecommunications, the globalization of markets and supply chains, the creation of more and more complex products, and so on. Many of humankinds' most powerful innovations, including bureaucracies, markets, judicial systems, democracy, modern logistics, concurrent engineering, and even the Internet represent, in essence. advances in our ability to coordinate. Coordination mechanisms have been studied in a wide range of disciplines, ranging from biology to economics to computer and organizational science, so it is likely that the typical participants in a BPR engagement will be unaware of many potentially useful coordination techniques. For all these reasons, coordination represents an excellent central focus for a process repository intended to support BPR.

While coordination represents a substantial body of knowledge, it is well within the realm of possibility to systematize it and we have made substantial progress in this regard. There are two main components to the coordination content in the Handbook repository. The first is the specialization hierarchy of coordination mechanisms, which includes roughly 200 "manage sharing" processes (see figure 13), 140 "manage fit" processes, and 400 "manage flow" processes. The actual scope of this hierarchy is in a sense larger than it appears, because in many cases (e.g. with auctions) a wide range of mechanisms is captured compactly as a single generic process model whose sub-activity specializations can be re-combined in many ways. The second major component is the taxonomy of coordination exceptions and associated handlers. The exception taxonomy includes over 400 exceptions, divided into three main categories: resource failures (where a resource breaks down), commitment violations (where an actor fails to discharge a commitment that it made e.g. to deliver a result on time) and emergent dysfunctions (where the aggregate effect of many agents acting in individually rational ways is in some way dysfunctional e.g. lynch mobs or stock market crashes). The specialization hierarchy of exception handler processes includes nearly 1000 entries.

Contributions

The Handbook BPR methodology differs in important respects from previous work. Previous efforts have been predicated on capturing a detailed as-is model, but provide little guidance concerning what the to-be process should look like. The innovativeness and quality of the new process depends, as a result, on the experience and the creativity of the particular individuals involved. In addition, because creating the detailed as-is model is so resource-intensive, the new idea generation phase is typical time-limited, so often only a few new ideas are evaluated in any depth. The Handbook methodology turns this on its head (Figure 18):

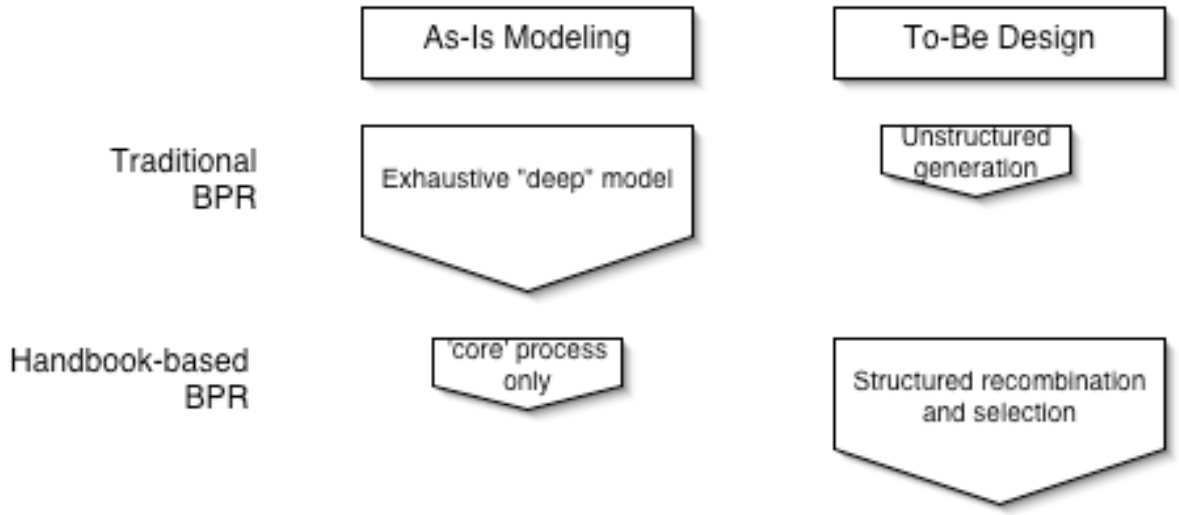

Figure 18. The Handbook BPR methodology compared with previous BPR approaches.

BPR participants are encouraged to capture only the "essence" of the process they wish to re-design, focusing on core processes and key dependencies. This step is relatively quick, and creates an as-is process model that is simpler and therefore easier to understand. The BPR participants spend the bulk of their time systematically exploring ideas inspired by (recombinations of) "best practices" harvested from many sources and industries. All elements of the deep structure, including those that "have always been done that way", are subjected to scrutiny. Designers are thus apt to consider a wider range of ideas than they would have generated on their own, and are more likely to take advantage of technological and managerial advances that have appeared in other contexts.

While others have explored the use of reusable process knowledge libraries to enable BPR (Committee 1992) (McNair and Leibfried 1992) (Schank and Abelson 1977) (Magazine 1992) (Mi and Scacchi 1993), (Salancik and Leblebici 1988) (Baligh, Burton et al. 1990) (Gasser 1992), our approach is unique in that it draws together a large repsotiry of process best-practices, organized in innovative ways using coordination theory concepts developed at MIT, and exploited using recombination-based methodologies derived from experience with re-designing processes of realistic scope and complexity.

Future Work

The generative strength of the Handbook methodology is, in a sense, a two-edged sword, in that it is often easy to uncover an overwhelming number of process alternatives. Imagine, for example, that we start with a deep structure model with 5 core activities and 5 alternative specializations per core activity. This produces 3125 (5^5) potential process alternatives, without considering alternatives for the key dependencies and important exceptions, if any. While many of these alternatives may be innovative and worthy of further exploration, many others will probably be unsuited to this particular domain, or even unworkable in general. The domain experts involved in the BPR engagement are thus called upon to be able to rapidly prune a large search space of possibilities so they can concentrate their effort on the most promising ones. While we do not expect to obviate the need for human judgment, we do plan to explore how the system can support human users by reducing the burden of traversing large design search spaces. One possibility involves the development of algorithms for

automatically exploring the process design space for high-utility alternatives. Another promising direction involves enriching the metrics information captured in the Handbook through the development of a taxonomy of process attributes. Such a taxonomy can help identify which metrics should be captured in the Handbook's process descriptions and tradeoff tables, as well as help process designers with benchmarking and the generation of what-if scenarios.

Acknowledgements

This paper would not have been possible without the many valuable contributions made by Youssef Bouali, Alessandro Margherita, Nezha Sadguy and others at ISUFI , as well as George Herman, Thomas Malone, John Quimby and others at MIT.

References

Armistead, Colin and Philip Rowland. 1996. *Managing Business Processes: BPR and Beyond.* New York: John Wiley and Sons.

Baligh, H. H., R. M. Burton, et al. 1990. Devising expert systems in organization theory: The Organizational Consultant. *Organization, Management, and Expert Systems*. M. Masuch. Berlin Germany, Walter de Gruyter: 35-57.

Bernstein, A., Klein, M., & Malone, T. W. 1999. The Process Recombinator: A tool for generating new business process ideas. *Proceedings of the International Conference on Information Systems*, Charlotte, NC, December 13-15.

Camp, Robert C. 1995. *Business Process Benchmarking : Finding And Implementing Best Practices* Milwaukee, Wis. : ASQC Quality Press.

Chen, Minder. 1999. BPR Methodologies: Methods and Tools. in D. Jack Elzinga, Thomas R. Gulledge, Chung-Yee Lee (Eds). *Business process engineering : advancing the state of the art.* Norwell, Mass. : Kluwer Academic Publishers, p. 187-212.

Committee, A. T. Q. S. 1992. *Benchmarking: Focus on World-Class Practices*. Indianapolis, IN USA, AT&T Bell Laboratories.

Crowston, K. G. 1991. *Towards a Coordination Cookbook: Recipes for Multi-Agent Action*, MIT Ph. D. Thesis, Sloan School of Management.

Davenport, T.H. and M.A. Perez-Guardado. 1999. Process Ecology: A New Metaphor for Reengineering-Oriented Change. In *Business Process Engineering : Advancing The State Of The Art*, T.R.G. D. Jack Elzinga, Chung-Yee Lee, Editor. Kluwer Academic Publishers: Norwell, Mass. p. p. 25-44.

Davenport, T. H. 1995. Business Process Reengineering: Where It's Been, Where It's Going, in V. Grover and W. J. Kettinger. eds.). In *Business Process Change: Concepts, Methods and Technologies*Idea Publishing, Harrisburg PA, 1-13.

Davenport, T.H. 1994. Reengineering: Business Change of Mythic Proportions? *MIS Quarterly*, pp. 121-127.

Davenport, T.H. & Short, J.E. 1990. The New Industrial Engineering: Information Technology and Business Process Redesign, *Sloan Management Review*, pp. 11-27.

Dellarocas, C. 1996. *A Coordination Perspective on Software Architecture: Towards a Design Handbook for Integrating Software Components*. Ph.D. Dissertation. MIT Dept. of Electrical Engineering and Computer Science, MIT. Cambridge, MA

Fillmore, C. 1968. The case for case. E. Bach and R. T. Harns. Eds. *Universals in Linguistics Theory.* Chicago, Holt, Rinehart and Winston**,.** 1968), 1-90.

Fillmore, C. 1975. *Principles of Case Grammar: the Structure of Language and Meaning*. Tokyo, Sanseido Publishing Company,

Fadel, F. G., Fox, M.S., Gruninger, M. 1994. A Generic Enterprise Resource Ontology, *Proceedings of the Third Workshop on Enabling Technologies - Infrastructures for Collaborative Enterprises* , West Virginia University.

Gasser, L. 1992. HITOP-A: Coordination, Infrastructure and Enterprise Integration. *AAAI-92 Workshop on AI in Enterprise Integration.*

Gersick, C. 1991. Revolutionary change theories: A multilevel exploration of the punctuated equilibrium *Academy of Management Review* 16(1. 10–36.

Giddens, A. 1986. *The Constitution of Society: Outline of the Theory of Structuration*. University of California Press, Berkeley, CA.

Gomez, P.-Y. and B. C. Jones. 2000. Conventions: an interpretation of deep structure in organization. *Organizational Science* 11(6): 696-708.

Grover, V. and W. J. Kettinger, Eds. 1995. *Business Process Change: Concepts, Methodologies and Technologies*. Harrisburg, Idea Group.

Guarino, N. 1998. Formal ontology in Information Systems. *Proceedings of FOIS'98,* IOS Press

Hammer, M. 1990. Reengineering Work: Don't Automate, Obliterate. *Harvard Business Review* 68, 4. 104-112.

Hammer, M. J. Champy. 1993. *Reengineering the Corporation: A Manifesto for Business Revolution.* New York NY USA, Harper Business.

Harkness, W. L., Kettinger, W. J. and Segars, A. H. 1996. Sustaining Process Improvement and Innovation in the Information Systems Function: Lessons from the Bose Corporation, *MIS Quarterly* 20(3) 349-368.

Jaarvenpaa, S. and Stoddard, D. B. 1998. Business Process Redesign: Radical and Evolutionary Change *J. Business Research* 41(1) 15-27

Kettinger, W. J., S. Guha, et al. 1997a. The process reengineering life cycle methodology: A case study. In V. Grover and W. J. Kettinger. Eds. *Business Process Change: Concepts, Methodologies and Technologies.* Idea Group**:** 211-244.

Kettinger, W. J; Teng, J.T.C. and Guha, S. 1997b. Business Process Change: A Study of Methodologies, Techniques, and Tools *MIS Quarterly*, 21, 1, 55-80.

Klein, M., G.A. Herman, J. Lee, E. O'Donnell, and T.W. Malone. 2003. Inventing New Business Processing Using a Process Repository. In *Organizing Business Knowledge: The MIT Process Handbook*, T.W. Malone, K. Crowston, and G.A. Herman, Editors. MIT Press: Cambridge MA USA.

Klein, M., C. Dellarocas. 2003b. Designing Robust Business Processes. In *Organizing Business Knowledge: The MIT Process Handbook*, T.W. Malone, K. Crowston, and G.A. Herman, Editors. MIT Press: Cambridge MA USA.

Klein, M. and A. Bernstein. 2004. Towards High-Precision Service Retrieval. *IEEE Internet Computing Journal*. 8(1): p. 30-36.

Kruschwitz, N. and G. Roth. 1999. *Inventing Organizations of the 21st Century: Producing Knowledge Through Collaboration*. MIT CCS Working Paper #.207

Kubeck, Lynn C. 1995. *Techniques for Business Process Redesign*, New York: John Wiley and Sons.

Kubeck, Lynn C. 1997. Techniques for Business Process Redesign. *Interfaces*. 27 (4) 82

Lee, J. & J., M. Grunninger, Y. Jin, T. Malone, A. Tate, G. Yost and other members of the PIF Working Group. 1998. The Process Interchange Format Framework. *Knowledge Engineering Review*, 13(1). Cambridge Univ. Press. pp. 91-120

Lee, J. and Pentland, B.T, 2000. *Grammatical approach to Organizational Design.* Cambridge, MA: MIT Sloan School of Management.

Lenat, D. B. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38 (11).

Magazine, C. 1992. Back Support for Benchmarkers. *CIO Magazine* June.16.

Malone, T. W. and K. Crowston. 1994. The interdisciplinary study of Coordination. *ACM Computing Surveys* 26(1) 87-119.

Malone, T. W., K.Crowston, J. Lee, B. Pentland, et al. 1999. Tools for inventing organizations: Toward a handbook of organizational processes. *Management Science* 45(3. pp.425-443

Malone, T.W., K. Crowston, and G.A. Herman, eds. 2003. *Organizing Business Knowledge: The MIT Process Handbook*. MIT Press: Cambridge MA USA.

McNair, C. J. and K. H. J. Leibfried. 1992. *Benchmarking: A Tool for Continuous Improvement*. New York NY USA, Harper Business.

Mi, P. and W. Scacchi. 1993. Articulation: An Integrated Approach to the Diagnosis, Replanning and Rescheduling of Software Process Failures. *8th International Conference on Knowledge-Based Software Engineering.*

Mi, P. and W. Scacchi. 1996. A Meta-Model for Formulating Knowledge-Based Models of Software Development. *Decision Support Systems*, 17(4):313-330,. 1996.

Nissen, M. 1998. Redesigning reengineering through measurement-driven inferences. *MIS Quarterly* 22(4).

Nissen, Mark E. 1999. A Configuration-Contingent Enterprise Redesign Model. in D. Jack Elzinga, Thomas R. Gulledge, Chung-Yee Lee.(Eds. *Business Process Engineering : Advancing The State Of The Art*. Norwell, Mass. : Kluwer Academic Publishers, p. 145-186.

Pandya, Vinodrai K, Nelis, S. 1998. Requirements for process redesign tools. *International journal of computer applications in technology*. 11(6): 409-418.

Pentland, B. T. 1995. Grammatical Models of Organizational Processes. *Organization Science,* 6 (5) 541-556

Salancik, G. R. and H. Leblebici. 1988. Variety and form in organizing transactions: A generative grammer of organizations. In *Research in the Sociology of Organizations*. N. DiTomaso and S. B. Bacharach. Greenwich, CT USA, JAI Press.

Schank, R. C. and R. P. Abelson. 1977. *Scripts, Plans, Goals And Understanding.* Lawrence Erlbaum Associates.

Schein, E. 1980. *Organizational Psychology*. Prentice-Hall, Englewood Cliffs, N.J.

Stefik, M. and S. Smoliar. 1995. Creative Mind; Myths and mechanisms; Six Reviews and A Response. *Artificial Intelligence* 79(1)

Stoddard, D. and Jarvenpaa, S. 1995. Business Process Reengineering: Tactics for Managing Radical Change, *Journal of Management Information Systems* 12 (1) 81-108.

Swartout, S. and A. Tate. 1999. Guest Editors' Introduction:Ontologies & Their Applications. IEEE Intelligent Systems 14(1).

Taivalsaari, A. 1996. On the notion of inheritance. *ACM Computing Surveys*. 28(3):438-479.

van der Aalst, W. M. P. and T. Basten. 1999. *Inheritance of Workflows: An approach to tackling problems related to change*. Technical Report Eindhoven University of Technology. Eindhoven,

Varun Grover, William J. Kettinger. 1995. *Business Process Change : Reengineering Concepts, Methods, And Technologies.* Harrisburg, PA : Idea Group Pub.

Winograd, T. 1981. *Language as a Cognitive Process*. Reading, MA: Addison-Wesley.

Wyner, G. and Lee, J. 2001. *Defining Specialization for Process Models*. MIT Sloan School Technical Report #4159.