THE PROCESS RECOMBINATOR: A TOOL FOR GENERATING NEW BUSINESS PROCESS IDEAS[1]

Abraham Bernstein
Mark Klein
Thomas W. Malone
Center for Coordination Science
Sloan School of Management
Massachusetts Institute of Technology
U.S.A.
{avi, m_klein, malone}@mit.edu

Abstract

A critical need for many organizations in the next century will be the ability to quickly develop innovative business processes to take advantage of rapidly changing technologies and markets. Current process design tools and methodologies, however, are very resource-intensive and provide little support for generating (as opposed to merely recording) new design alternatives.

This paper describes the Process Recombinator, a novel tool for generating new business process ideas by recombining elements from a richly structured repository of knowledge about business processes. The key contribution of the work is the technical demonstration of how such a repository can be used to automatically generate a wide range of innovative process designs. We have also informally evaluated the Process Recombinator in several field studies, which are briefly described here as well.

Keywords: process innovation, business process repository, BPR, business process design

1. THE CHALLENGE: DESIGNING INNOVATIVE PROCESSES

Most management observers today agree that the successful organizations of the 21st century will need to be able to develop new business processes more rapidly than they have in the past. In order to take advantage of rapidly changing markets and technologies, companies will need to continually keep developing new processes and new ways of using technology.

But where will the ideas for these new processes come from? Today's business process design tools provide little or no support for generating innovative business process ideas. The available tools are

[1] To appear in the Proceedings of the International Conference on Information Systems, 1999
Bernstein, Abraham, Mark Klein, and Thomas W. Malone. 1999. "The Process Recombinator: A Tool for Generating New Business Process Ideas." in *ICIS*. Charlotte, NC. (Nominated for Best Paper.)

primarily limited to *recording* existing processes in some formal representation (e.g. flowcharts) or to *analyzing* proposed processes (e.g., using quantitative simulations).

Today's business process designers, therefore, must rely almost entirely on their own intuitions and experience to generate new process ideas. The typical result of this situation is that relatively few alternatives are generated, and the ideas that do emerge often tend to be quite similar to practices already familiar to the designers (Ulrich and Eppinger (1995), p. 79).

This paper describes a new approach to this problem, based on the notion that most new ideas are often novel combinations of elements that already exist in some form. The key idea of our approach is that a richly structured on-line repository of knowledge about business processes can significantly enhance the creativity of process designers by helping them systematically explore many alternative combinations of process elements. Such an approach could, of course, be used with purely random combinations of process elements. However, by structuring the knowledge repository using a rich network of empirically-based process templates, we greatly increase the likelihood that useful alternatives will be generated.

We call the tool that implements this combinatorial approach to process innovation the 'Process Recombinator.' We built the Process Recombinator as an add-on to the MIT Process Handbook (Malone et al. (1999)). The next section provides a brief overview of the Process Handbook and the theoretical concepts upon which it is based. Then, Section 3 describes the Process Recombinator itself, and illustrates its use with examples from a field study we conducted. Finally, Section 4 evaluates the contributions this work has made and concludes with a discussion of possible directions for future research.

2. THE PROCESS HANDBOOK

The Process Handbook has been under development at the MIT Center for Coordination Science for over six years, including the contributions of a diverse and highly distributed group of over 40 university researchers, students and industrial sponsors (see the Appendix of this paper and Malone et al. (1999) for more detailed descriptions). The goal of the Handbook is to develop a repository and associated conceptual tools to help users effectively retrieve and exploit the process knowledge relevant to their current challenges. Currently the project focuses on the repository's application to business process re-design, sharing knowledge about business processes, and automatic software generation. The current repository has over 5000 process descriptions ranging from specific examples (e.g., a Mexican beer factory, an automobile parts manufacturer, and a university purchasing department) to more generic templates (e.g. for logistics, concurrent design, resource allocation and decision techniques). A Windows-based tool for editing the Handbook repository, as well as a Web-based tool for viewing it have been developed (Bernstein et al. (1995)). We have applied the Handbook to process re-design in collaboration with a management consulting firm and others. The successful outcomes of these experiences led to the development of the Process Recombinator.

The Process Handbook takes advantage of two simple but powerful theoretical concepts to organize process knowledge: *process specialization*, and the notion of *dependencies and their coordination mechanisms*.

2.1. Process Specialization

Practically all process representation techniques (including ours) use the notion of decomposition: that a process can be broken down (or "decomposed") into subactivities. Our representation includes in addition to this the concept of ‘specialization.’ While a subactivity represents a *part of* a process; a specialization represents a *type of* (or way of doing) the process.

Using this concept, processes can be arranged in a hierarchical structure with very generic processes at one extreme and increasingly specialized processes at the other. As in object-oriented programming, the specialized processes inherit properties of their more generic "parents," except where the specialized processes explicitly add, delete or change a property. Unlike traditional object-oriented programming, however, our inheritance is organized around a hierarchy of increasingly specialized processes (verbs) not objects (nouns).

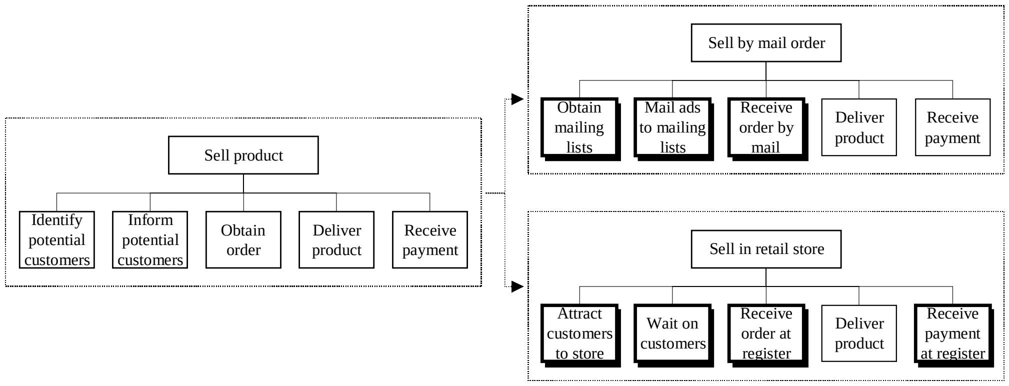


Figure 1: An example of inheritance in the specialization hierarchy (changed subactivities are shadowed)

The generic activity called "Sell product", for example, can be *decomposed* into subactivities like "Identify potential customers" and "Inform potential customers” (illustrated in Figure 1). It can also be *specialized* into variations like "Sell by mail order" and "Sell in retail store." These specialized activities inherit many subactivities from their "parent" process, but also typically include changes as well. “Sell in retail store”, for instance, replaces “Identify potential customers” by the more specialized activity “Attract customers to store.”

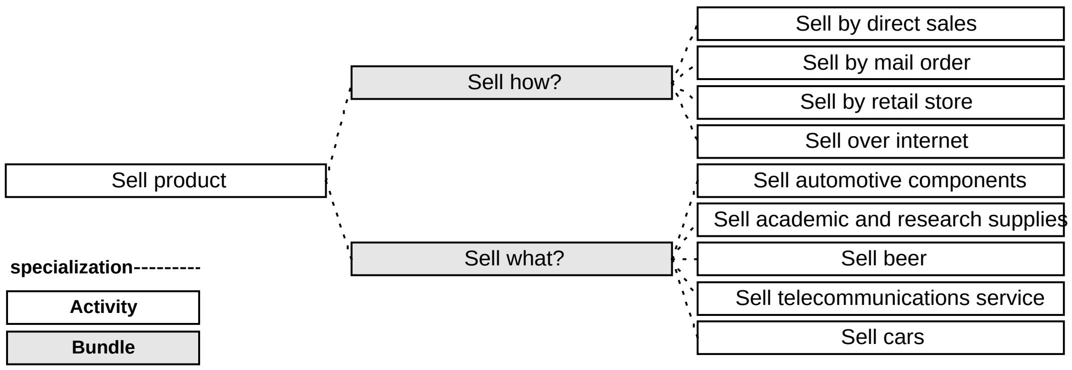
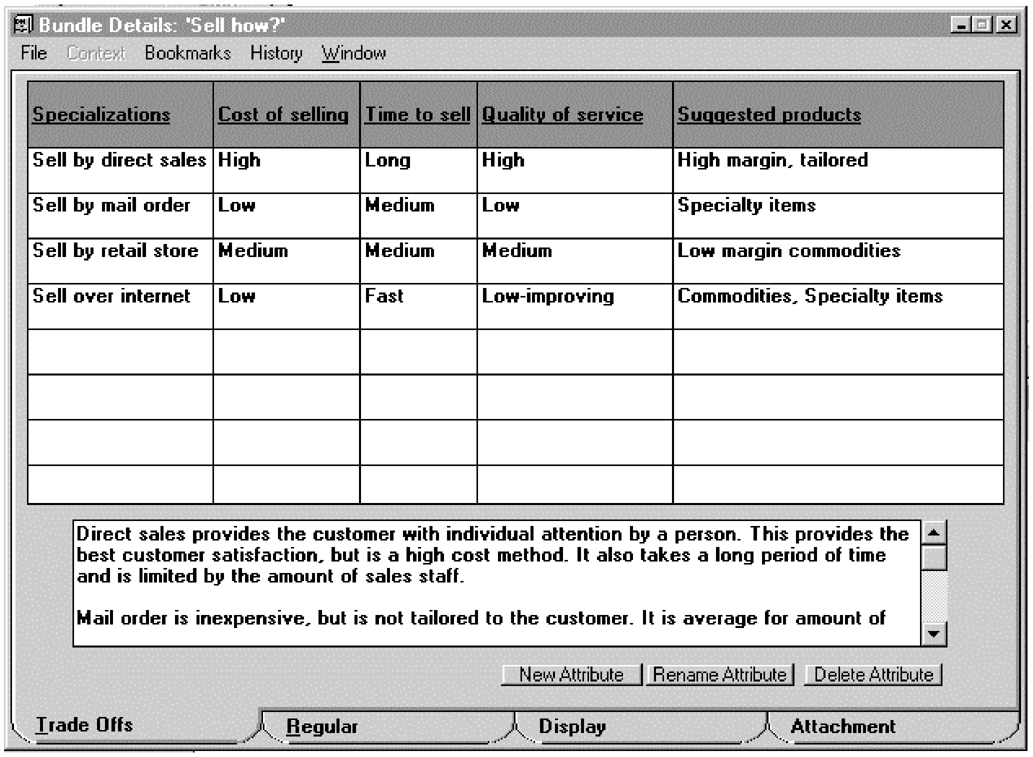


Figure 2: An example of bundles in the specialization hierarchy.

We have found it useful to group specializations into what we call "bundles" of related alternatives. Figure 2 gives two examples of such bundles in the specialization hierarchy for the 'Sell product" process. One bundle ("Sell how?") collects alternatives for how the sale is made (direct mail, retail storefront, or electronically), while the other ("Sell What?") concerns what is being sold (beer, automotive components, etc.) Generally speaking, bundles represent different dimensions along which processes can be classified.

Bundles can also have associated tradeoff tables that capture the relative pros and cons of the alternative 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:



Specializations	Cost of selling	Time to sell	Quality of service	Suggested products
Sell by direct sales	High	Long	High	High margin, tailored
Sell by mail order	Low	Medium	Low	Specialty items
Sell by retail store	Medium	Medium	Medium	Low margin commodities
Sell over internet	Low	Fast	Low-improving	Commodities, Specialty items

Direct sales provides the customer with individual attention by a person. This provides the best customer satisfaction, but is a high cost method. It also takes a long period of time and is limited by the amount of sales staff.

Mail order is inexpensive, but is not tailored to the customer. It is average for amount of

Figure 3: An example of a tradeoff table.
(Note that these particular values are for illustrative purposes only)

Entries in tradeoff tables can be generated by academic research, empirical observation, or soliciting the judgment of specialists in the process domain.

A specialization tree so structured can be viewed as a decision tree. If users want to find a process with given attributes, they can traverse from the root, and at each bundle select the one or more branches that seem to match what they are looking for. This property will prove important below when we use the specialization tree to support process (re-)design.

2.2. Dependencies and Coordination Mechanisms

The second key concept we use is the notion from coordination theory that coordination can be viewed as the management of task *dependencies* each managed by their own *coordination mechanism* (see Malone and Crowston (1994)). Dependencies arise from resources (e.g. parts, documents, signals) that are used by multiple activities. We typically analyze dependencies using three elementary dependency types: flow, sharing and fit (Crowston (1991), Zlotkin (1995). See Figure 4). Flow dependencies arise whenever one activity produces a resource that is used by another activity. 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 a single resource (e.g. when several designers create subcomponents for a single system).

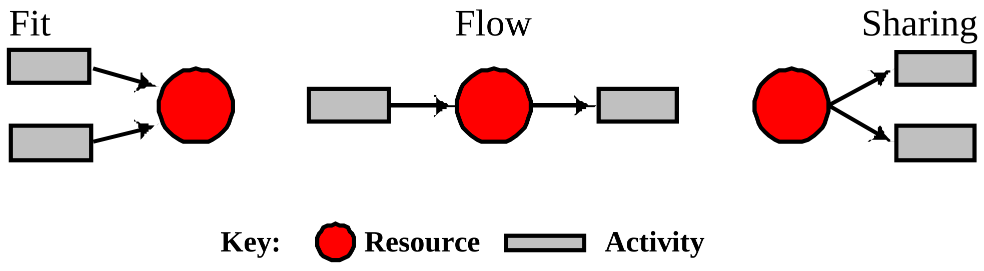


Figure 4: Three basic types of dependencies among activities.

The relationships represented by dependencies are managed by processes called coordination mechanisms. There is a wide range of coordination mechanisms potentially applicable for each kind of dependency.

Dependency	***Examples of coordination mechanisms for managing dependency***
Flow	
Prerequisite ('right time')	Make to order vs. make to inventory ("pull" vs."push"). Place orders using "economic order quantity," "Just In Time" (kanban system), or detailed advanced planning.
Accessibility ('right place')	Ship by various transportation modes or make at point of use
Usability ('right thing')	Use standards or ask individual users (e.g., by having customer agree to purchase and/or by using participatory design)
Sharing	"First come/first serve," priority order, budgets, managerial decision, market-like bidding.
Fit	Boeing's total simulations. Microsoft's daily build.

Table 1. Examples of dependencies and associated coordination mechanisms.

Managing a flow dependency, for example, usually involves making sure that the right thing (usability) arrives at the right place (accessibility) at the right time (timing). A tire-supplier and a car-manufacturer, for example, have a flow dependency between them. This flow consists of three parts: First tires have to be transported from the supplier to the manufacturer (i.e. tires have to be made *accessible* to the car-manufacturer. Second the tires have to be manufactured before they can be attached to the car (i.e. the tires have to arrive at the '*right time*'). Finally the tires delivered actually have to match the car (i.e. their *usability* has to be ensured). A flow is managed, when all of those parts are managed. To manage the prerequisite part, for example, Table 1 shows two possible coordination mechanisms: make-to-order (a variant of which is called just-in-time production) and make-to-inventory (where a stockpile of the product is created in anticipation of future demand). These mechanisms can be applied to almost any domain.

An example of a sharing dependency is the management of room usage. An often-used mechanism for ensuring the management of this dependency is ‘first come/ first serve’. As soon as someone signs up for use of the room it is booked. However most other processes for managing scarce resources could be applied to managing the room’s usage.

Microsoft’s daily build (see Cusumano and Selby (1995)) is an excellent illustration of managing a fit dependency. Given the need of components of the operating system to fit each other Microsoft chooses to ensure their compatibility by forcing their co-functioning in a daily build, which is then used as a baseline the next day.

3. THE PROCESS RECOMBINATOR

The Process Recombinator is a software tool that uses the Process Handbook to support a process innovation methodology based on the concepts described above. This methodology consists of three key steps (Herman et al. (1998) contains a detailed explanation of the methodology):

1. *identify* the core activities and the key dependencies (i.e. the *deep structure*) of the process you want to redesign, using the process specialization hierarchy
2. systematically *generate* a set of alternative refinements (i.e. *surface structures*) for the tasks and dependencies in this deep structure model, by ‘recombining’ existing or newly generated alternatives for these process components
3. *select* from this set the process(es) that appear to best satisfy your requirements, possibly using information stored in tradeoff matrices

We will describe how these steps are accomplished in the sections below. The capabilities underlying step 1 and 3 are part of the original set of Process Handbook tools, so we will summarize them quickly and focus the bulk of the paper on the Recombinator capabilities of step 2.

To illustrate the capabilities of the Recombinator tool, we will use examples based on a field study we conducted in collaboration with one of our corporate research sponsors, the AT Kearney consulting firm, and one of their clients which we call “Firm A” to preserve the client’s anonymity (see Malone et al. (1999), Herman et al. (1998); and Kruschwitz and Roth (1999) for more detailed descriptions of this study).

Firm A was experiencing increasing problems with their hiring process. They were growing rapidly in a tightening labor market, and they had a culture of independent, competitive business units. Together, these factors led to increases in the time and cost to hire people and to increasingly frequent instances of business units “hoarding” candidates or bidding against each other for the same candidate. In an effort to improve their hiring process, the organization had invested a great deal of time and energy into “as is” process analysis using conventional techniques such as flowcharting. But they also wanted some way to come up with highly innovative ideas about how to improve their process.

The Recombinator was completed after the field study, and the examples shown here demonstrate how the tool now supports the manual process that was followed in the field study.

3.1. Identifying the Process Deep Structure

The first step in our methodology is to identify the *deep structure*, i.e. a process model which captures the essence (i.e. the core activities and key dependencies) of the process we wish to redesign. This maximizes room for new ideas by abstracting away non-essential features. The Handbook supports this via the specialization hierarchy. Users can either select an existing generic process from the hierarchy or create a new one. Since Firm A wanted to improve their hiring process, we use the "Hire" process as the starting point for our example scenario (see Figure 5).

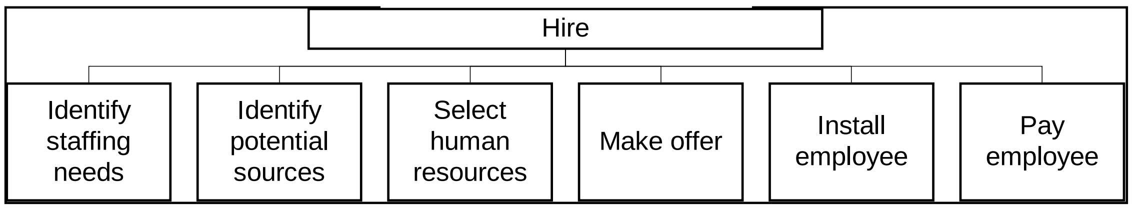


Figure 5: The Deep Structure for "Hire"

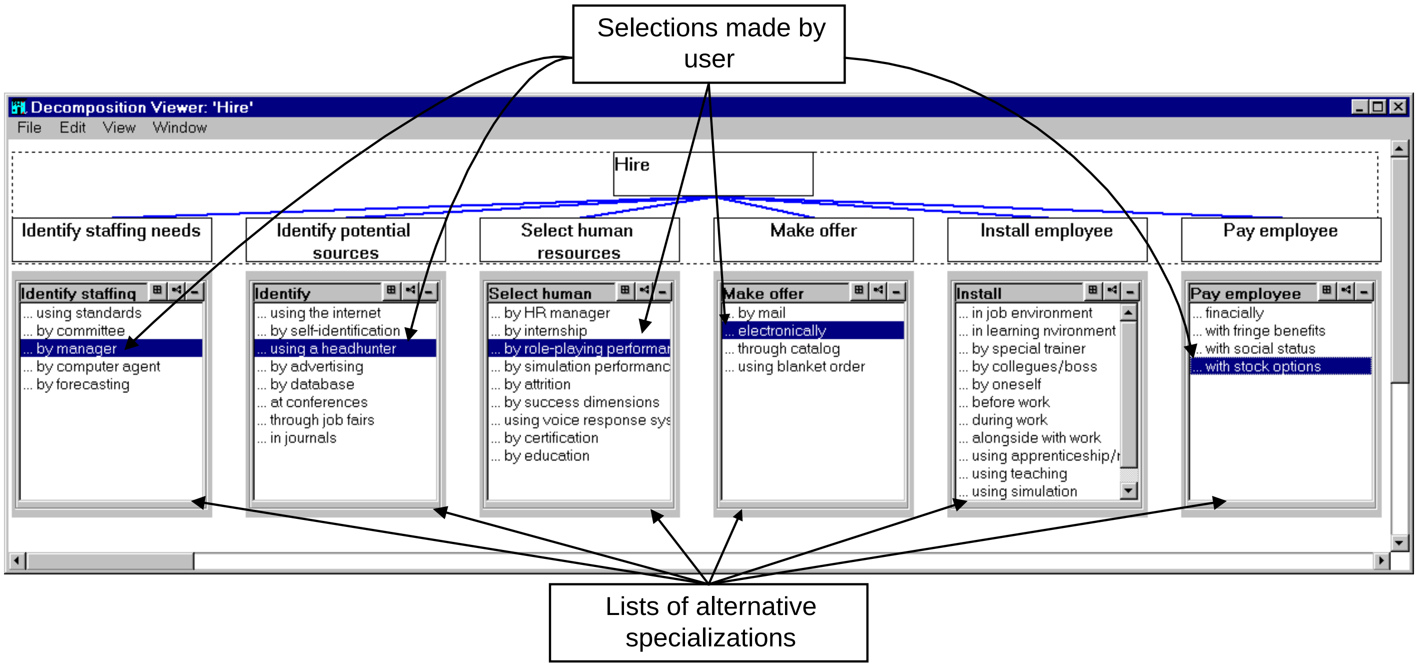
3.2. Process Recombination

The next step is to find alternative ways (i.e. different *surface structures*) for implementing the generic activities and coordination mechanisms identified in the deep structure model. This is achieved by the Process Recombinator itself, which includes three parts. The three parts can be used independently; each allows systematic exploration along a different set of process design dimensions. First, we will look at the *subactivity recombinator*, which generates all possible combinations of the specializations of the subactivities in the process. Next, we consider the *dependency recombinator*, which generates different combinations of coordination mechanisms for the process dependencies. Finally, we will look at the *bundle recombinator*, which generates different combinations of the alternatives in the dimensions represented as bundles.

Please note that the order of usage was chosen for illustrative purposes only. The three parts of the Recombinator can be used in different sequences depending on one's needs.

The Subactivity Recombinator

The Subactivity Recombinator lets users pick different specializations for each of the subactivities in a process (see Figure 6). For example, the "Select human resources" subactivity of the hiring process has specializations such as (a) "Select by role-playing" (a process used, for instance, by Cessna to screen candidates for executive positions), (b) "Select based on education" (a screening process implicitly used by many management consulting firms), and (c) "Select by attrition" (a screening process used by universities who admit all applicants and then fail many of them in the first year). Using the Process Handbook capabilities, users can easily see more detailed descriptions (and other information) about each of these activities.


Selections made by user
Lists of alternative specializations
Decomposition Viewer: 'Hire'
File Edit View Window

Hire

Identify staffing needs	Identify potential sources	Select human resources	Make offer	Install employee	Pay employee
... using standards	... using the internet	... by HR manager	... by mail	... in job environment	... finacially
... by committee	... by self-identification	... by internship	... electronically	... in learning nvironment	... with fringe benefits
... by manager	... using a headhunter	... by role-playing performar	... through catalog	... by special trainer	... with social status
... by computer agent	... by advertising	... by simulation performanc	... using blanket order	... by collegues/boss	... with stock options
... by forecasting	... by database	... by attrition		... by oneself	
	... at conferences	... by success dimensions		... before work	
	... through job fairs	... using voice response sys		... during work	
	... in journals	... by certification		... alongside with work	
		... by education		... using apprenticeship/r	
				... using teaching	
				... using simulation	

Figure 6: The Subactivity Recombinator user interface.

As the figure illustrates, each subactivity is shown in a different column, and each column contains the alternative specializations for that subactivity. Using this display, users select the combination of specializations they want to use in creating a new process. The system then automatically generates the new process specified. If users make multiple selections in some of the columns, then all combinations are generated. Figure 7 shows, as an example, the process created for the selections made in Figure 6. (Users who want to know more about how the alternatives in a given column compare can also click on the 'tradeoff' button for the column and see a tradeoff matrix for those alternatives.)

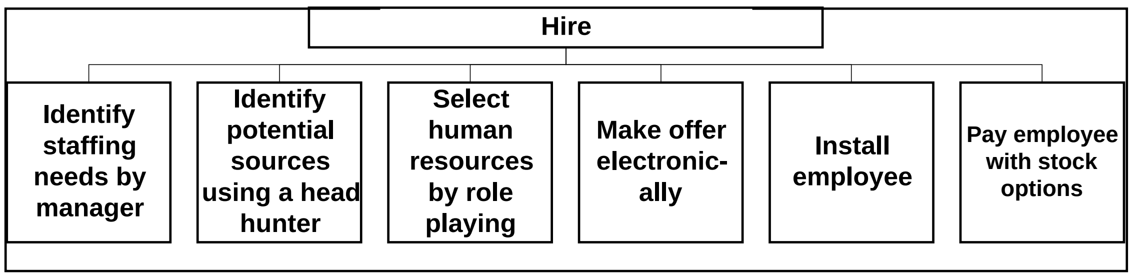

Hire
Identify staffing needs by manager
Identify potential sources using a head hunter
Select human resources by role playing
Make offer electronic-ally
Install employee
Pay employee with stock options

Figure 7: Results of using the Subactivity Recombinator.

The power of this approach is that the specialization hierarchy allows the process designer to draw on relevant ideas and insights from many different kinds of organizations, opening the possibility of useful combinations never before considered in this particular setting.

The Dependency Recombinator

The Dependency Recombinator complements the Subactivity Recombinator by allowing one to also consider alternative coordination mechanisms for process dependencies. Instead of displaying only the subactivities of a process, it displays both the subactivities and dependencies as a flow-chart (Figure 8):

Every subactivity and dependency can have an associated list of alternative choices below it. The lists below dependencies allow users to select the coordination mechanisms used to manage them. In Figure 8, for example, we can see different alternatives for managing the dependency between “Use headhunter for sourcing” and “Select human resources…” There could be a traditional “push-based” coordination, where the head hunter contacts the firm. Alternatively there could be an “open market” of ‘sellers’ (headhunters and internal HR-departments) and buyers (line-function departments). The “market with bonus” coordination mechanisms reimburses the seller (in our case the headhunter) with a fee depending on the new employee’s performance in the firm. This encourages headhunters to think about the long-term performance of a candidate. Once the user has selected alternatives for each subactivity and dependency, the system automatically generates new process designs in the same way as the Subactivity Recombinator.

By bringing in coordination possibilities from far afield (such as on-line bidding systems for internal recruiting), this approach can generate very innovative process possibilities.

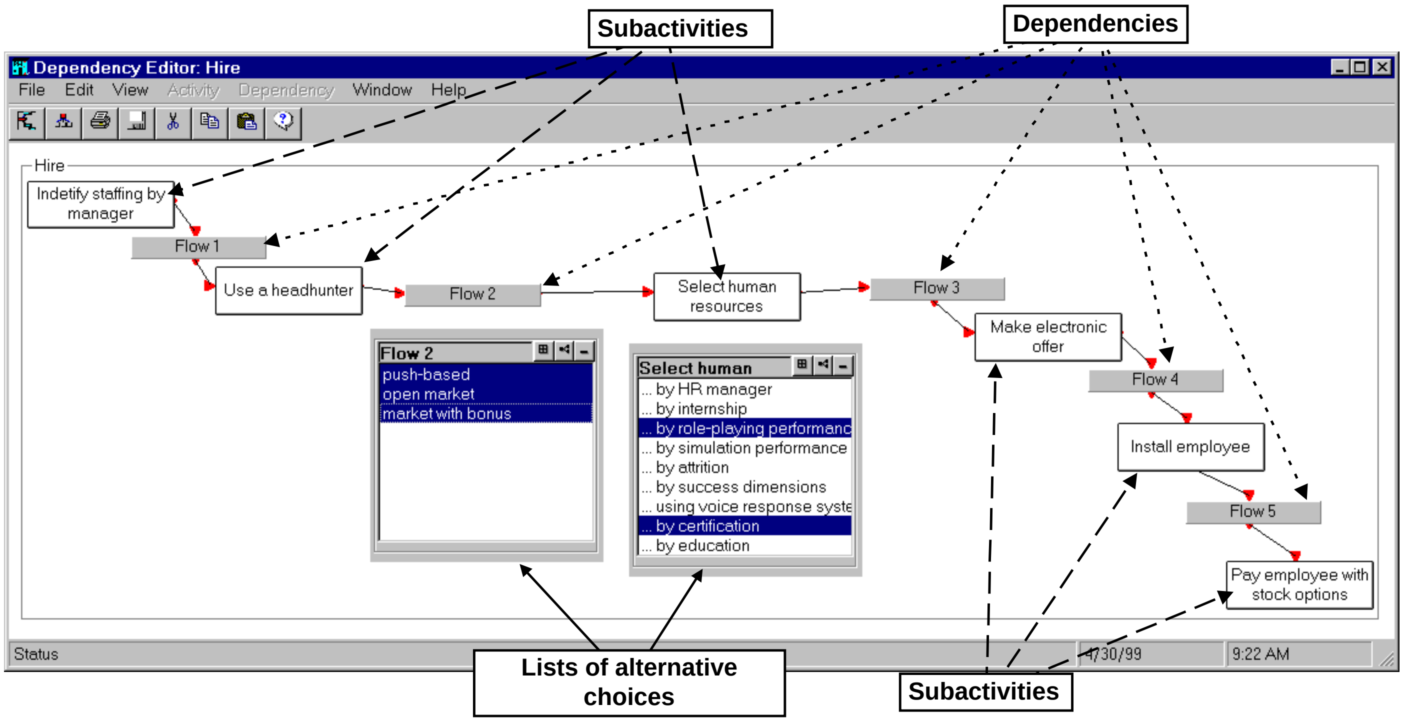


Figure 8: The Dependency Recombinator user interface.

The Bundle Recombinator

The Bundle Recombinator helps users generate new design alternatives by exploring the multiple possibilities defined by bundles in the specialization hierarchy. Consider, as an example, the specialization sub-tree under ‘Install Employee’[2] (Figure 9):

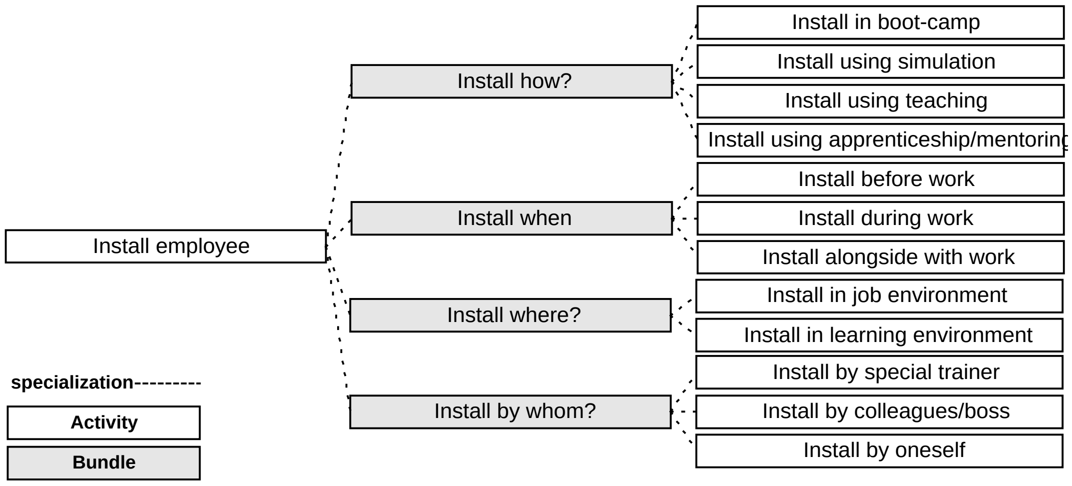


Figure 9: The specialization sub-tree for ‘Install employee’.

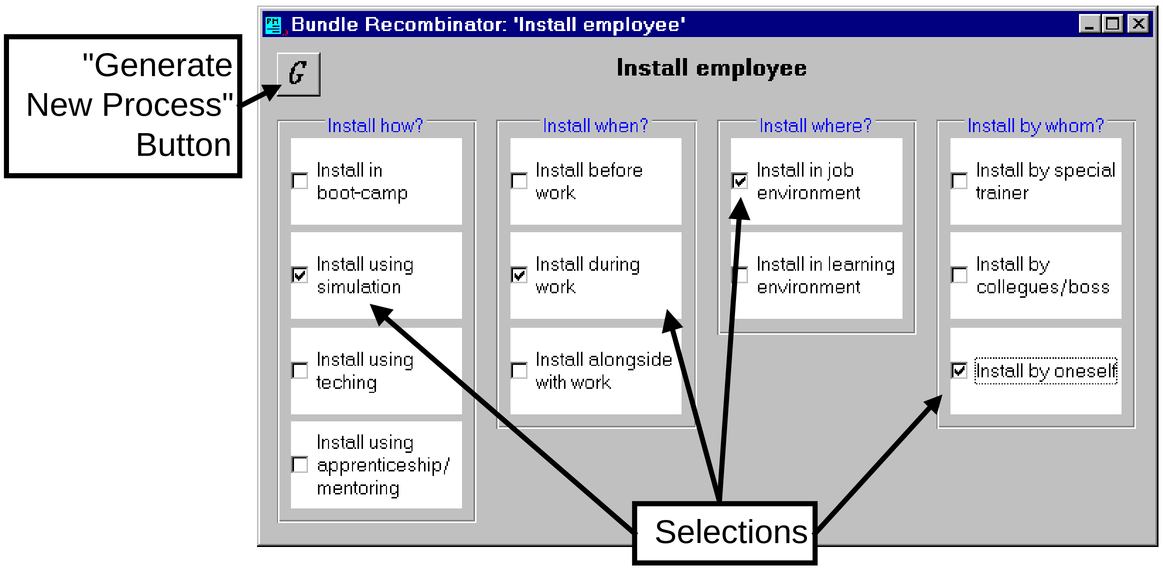
Recall that the bundles under a given process in the specialization hierarchy group together refinements of that process that differ along a particular dimension such as who does the work, or how it is done. Generally, each bundle captures an orthogonal design dimension. The four bundles under ‘Install employee’ therefore define a four dimensional space of possible combinations (for example Figure 10 shows the combinations defined by two of these dimensions):

		Install when?		
		Install before work	Install during work	Install alongside with work
By Whom?	Install by special trainer	▨	▨	▨
	Install by colleagues/boss	▨	▨	▨
	Install by oneself	▨	*	▨

Figure 10: Part of the design space for the ‘Install employee’ process (the cell marked is the example described in the text)

[2] The term “install” may seem mechanistic when applied to employees. The term itself, however, suggests potentially innovative analogies with situations where other kinds of things are “installed”.

Each cell in this four-dimensional space represents a possible new process specialization formed by making one selection from each bundle dimension. For instance, Figure 11 shows the combination “Install by oneself during work within the job environment.” One example idea stimulated by this combination is training novice air traffic control officers by interleaving simulations of unusual situations in the middle of their real work environment (perhaps without the trainees even knowing that these were simulations).


"Generate New Process" Button
Bundle Recombinator: 'Install employee'
G
Install employee
Selections

Install how?
- ☐ Install in boot-camp
- ☑ Install using simulation
- ☐ Install using teching
- ☐ Install using apprenticeship/ mentoring

Install when?
- ☐ Install before work
- ☑ Install during work
- ☐ Install alongside with work

Install where?
- ☑ Install in job environment
- ☐ Install in learning environment

Install by whom?
- ☐ Install by special trainer
- ☐ Install by collegues/boss
- ☑ Install by oneself

Figure 11: The Bundle Recombinator user interface.

Another interesting combination (stimulated by the combination of two dimensions marked by a “*” in Figure 10) is to let new employees “install” themselves by having them decide what they could do best for the firm. In this process, new employees look around to find something useful to do for the firm (Kaftan and Barnes (1991) report this type of behavior in some of the hires at SUN Hydraulics).

When users have selected a combination of alternatives (like that shown in Figure 11), they press the button shown in the upper left corner of the figure, and the system generates the new process they have specified. The subactivities in the newly created process are derived by “multiple inheritance” from the ‘parent’ processes (Wyner and Lee (1995)). The specific algorithm used for multiple inheritance is as follows: Subactivities that appear in one or both specialization parents are inherited as is. If one parent process has a more specialized form of a subactivity than the other does, then the more specialized version of that subactivity is inherited. If one parent process has deleted a subactivity that appears in another, or if a subactivity is specialized in different ways in the different parent specializations, then the system asks the user what to do.

The power of this approach is that it can suggest novel combinations that lead process designers to creative new ways of doing the process.

The roles of the different recombinators

As we have seen, the Subactivity and Dependency Recombinators have similar functionality, while the Bundle Recombinator takes an orthogonal approach. All three approaches can be used in a fully integrated way. The Subactivity Recombinator is useful when we wish to focus on alternatives for the core activities in the process. The Dependency Recombinator is useful when we wish to also explore different ways of managing the key dependencies in the process. The Bundle Recombinator, finally, allows one to create new process specializations suggested by using bundles as design dimensions. The new specializations created by any Recombinator tool can then, of course, be used as alternatives within the other ones. The decision of which Recombinator to use first is dependent on what aspect of a process seems to be most promising for generating novel processes. Exploring the design space of process ideas becomes an iterative process in which the three parts of the Recombinator are used in turn, until a satisfactory set of interesting alternatives is generated.

3.3. Comparing the new process designs

Once users have used the different components of the Process Recombinator to produce a number of candidate process re-designs, they can use a tradeoff matrix to help assess each re-design from the perspective of the criteria that are meaningful to them. The selections made in Figure 8 above, for example, would yield the rows shown in the tradeoff matrix in Figure 12.

The Handbook specialization hierarchy can include, for each process, attributes and associated values that describe the process. Attributes that are potentially appropriate for comparing the newly generated process alternatives are thus automatically inherited by the new combinations just as subactivities are. Thus, the columns of this trade-off matrix are also automatically generated by the system. In some cases, default values for the cells will be inherited as well. It is up to the user, however, to determine whether the values of these attributes are appropriate for each alternative and to change them if necessary. Once this is done, the new processes and associated tradeoff values are maintained in the Handbook repository as a source of ideas for future users.

	Cost of recruiting	Time horizon for recruiter	Complexity of selection
Hire using headhunter (push-based) select by certification	Hi	Short	Low
Hire using head hunter (open-market) select by certification	Medium	Short	Low
Hire using head hunter (market with bonus) select by certification	Medium	Long	Low
Hire using headhunter (push-based) select by role-playing performance	Hi	Short	High
Hire using head hunter (open-market) select by role-playing performance	Medium	Short	High
Hire using head hunter (market with bonus) select by role-playing performance	Medium	Long	High

Figure 12: Tradeoff Matrix for new process re-designs.
(All values are for illustration purposes only)

4. CONTRIBUTIONS OF THIS WORK

We view the primary contribution of this work to be the technical demonstration of how a richly structured repository of process examples can be used to automatically generate a wide range of ideas for innovative process designs. We have also, however, informally evaluated the Process Recombinator, and the methodology underlying it, in several 'real-life' contexts.

4.1. Informal evaluation based on field studies

The most substantive example to date has been the field study to re-design the hiring processes for Firm A used as the basis for the examples above (Herman et al. (1998)). Though this was not, by any means, a controlled experiment, the participants in this process innovation effort found the methodology and Handbook repository to be very effective in helping them generate a wide range of novel and promising process design alternatives. The Process Recombinator was completed after this field study in order to provide computational support for what we understood to be the key components of the methodology. We have re-enacted portions of these re-design experiences using the Recombinator and have found that it is effective in supporting them.

Since then the Recombinator has been used in another field study to support a large bank's efforts to design new distribution/sales processes for physical financial products like travelers' checks, foreign currency, and precious metals. Before the study, the bank had close to a hundred different processes of this type. After analyzing their processes, they realized that all the processes were well-captured as a set of bundles representing such dimensions as type of good, type of trading partner (bank, central bank, person), payment method, payment currency, and internal booking type. Once they had the space so systematized, they were able to use the recombination methodology to identify innovative combinations along these dimensions.

These examples demonstrate well both sources of power of the recombinator approach: (1) the way it allows process designers to draw from ideas drawn from many different organizations and domains, and (2) the spur to creative process design provided by the systematic generation of novel combinations of these 'best practice' ideas. Those examples thus support the notion that a rich repository of appropriately organized process templates, supported by tools like the Recombinator, can be of significant practical use in enhancing the creativity and efficiency of process innovation.

4.2. Comparison to related process design tools

We believe, therefore, that the Process Recombinator fills an important gap in existing process design technologies. As noted above, current techniques (Hammer and Champy (1993), Grover and Kettinger (1995), Harrington (1991), Kettinger, Guha and Teng (1995), Kettinger and Grover (1995), Davenport (1993)) offer little support for identifying new processes (Kettinger, Teng and Guha (1997)). They suggest how organizations can *organize* their process definition efforts (e.g. using brainstorming, visioning, meeting facilitators) as well as *record* the resulting process designs (e.g. using IDEF or Petri Nets), but do not help us actually *generate* new process alternative ideas.

Others have explored the use of re-usable process templates (AT&T Quality Steering Committee (1992), McNair and Leibfried (1992), Schank and Abelson (1977), CIO Magazine (1992), Mi and Scacchi (1993)), abstract process models (Sacerdoti (1974), Nau (1987)) and systematic process alternative generation (Salancik and Leblebici (1988)). Our work is unique, however, in how it systematically uses a large repository of empirically-based examples to systematically generate many alternative combinations of process elements.

Also related to our work are systems that automatically generate organizational designs based on descriptions of the organizational tasks and other factors (Baligh, Burton and Opel (1990), Gasser (1992), Majchrzak and Gasser (1992)). Baligh et al. for example, edited ‘textbook’ knowledge about organizational design into an ‘expert system’ that will make recommendations based on rules like ‘If the environment is stable, then a formal organization is appropriate’. Our work differs from these approaches in at least two ways: (1) We are interested not only in providing ‘conventional’ guidance for ‘traditional’ organizations, but also in providing tools to help ‘invent’ new organizations. (2) We are not attempting to provide completely automated advice based on simple input parameters (the traditional ‘expert systems’ approach). Instead, we are attempting to provide conceptual frameworks and partly automated tools to help enhance people’s abilities to creatively define and systematically explore a large process design space. That is, we want to provide a helpful tool for use by human experts, not an ‘automated expert’ that tells humans what to do.

5. FUTURE WORK

One of the major limitations of the Recombinator is the availability of a sufficient underlying knowledge base of processes. The size of the underlying knowledge base has a direct influence on the usefulness of the Recombinator. Our experience with the Process Handbook has shown that the current knowledge base of more then 5000 processes is adequate to generate interesting processes in a variety of domains. We believe that the universality of some of the concepts used (like the notion of the coordination mechanisms or the ubiquity of logistics processes) allow the tool to support some innovation with little prior content in the knowledge base. However future research about the applicability of the tool in different domains is needed.

In the future we plan to evaluate and refine the Recombinator in other domains (including logistics and manufacturing), and extend it to cover other aspects of our process innovation methodology such as generating new processes by sub-activity re-ordering. Another issue we would like to address concerns managing the size of the process design space. The generative strength of our approach is a two-edged sword in the sense that it is often easy to create an overwhelming number of process alternatives. The procedures are not fully formalized, so human judgment is often needed, for example, to select, from the candidates generated by the Recombinator, the process design appropriate for one’s needs. While we do not expect to obviate the need for human judgment, we do plan to explore how the Recombinator can further reduce the burden of exploring/pruning a large process design space. In the meantime, we believe that it will often preferable to have ‘too many’ options rather than too few.

6. ACKNOWLEDGEMENTS

We would like to thank George Herman and the other members of MIT Center for Coordination Science, as well as Elisa O’Donnell of A.T. Kearney Consulting, for their invaluable contributions to the ideas underlying this paper.

Section 2 of this paper was adapted from parts of Malone et al. (1999).

This material is based upon work partially supported by the National Science Foundation under Grant No. 9224093. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

7. APPENDIX: IMPLEMENTATION OVERVIEW

Software Implementation

The Process Handbook software provides a standard set of tools to browse, manipulate and store process descriptions. Our current system is implemented under the Microsoft Windows operating system using Microsoft's Visual Basic programming language and numerous third-party modules for that environment (i.e., COM/ActiveX-objects). The process descriptions are stored in a relational database (currently Microsoft Access) with an interface layer above the database that represents processes using the concepts described above (Ahmed (1998), Bernstein et al. (1995)). This interface, implemented as a COM-object, allows programs to retrieve, edit and add process descriptions. A number of different viewers and editors, including a web-based browser (see http://process.mit.edu), have been implemented as part of the Process Handbook Project (for more information see Malone et al. (1999)).

The Process Recombinator is an extension to the Process Handbook software. Using the same development environment we extended the existing tools to provide a user-interface for specifying recombinations. Once specified, the Recombinator accesses the Process Handbook database through the same interface-layer as the other viewers and editors and generates the new process models directly into the database. Those new models are then available for retrieval and manipulation using all of the tools provided by the Process Handbook software.

Content

The Process Recombinator software accesses the process knowledge base provided by the Process Handbook project. Numerous contributors developed content for the Process Handbook knowledge base, and the content was added to the knowledge base using the tools described above. The current repository has over 5000 process descriptions ranging from specific examples to more generic templates.

The contents of the Handbook come from both primary sources (such as student thesis projects) and secondary sources (such as published descriptions of innovative business practices). So far, we have focused our data collection on the domain of "supply chain management" – the process by which an organization (or group of organizations) manages the acquisition of inputs, the successive transformations of these inputs into products, and the distribution of these products to customers. For example, the handbook includes results from several MIT master's thesis studies of supply chain processes ranging from a Mexican beer factory to a university purchasing process (Geisler (1995), Leavitt (1995), Lyon (1995), Ruelas Gossi (1995)). The entries also include a number of examples drawn from the "Interesting Organizations Database" collected from published sources and student projects as part of an MIT research initiative on "Inventing the Organizations of the 21st Century." Furthermore, we have included processes from field-studies that the center or its sponsors undertook.

Finally, we have developed a framework of generic process descriptions. To develop such a framework, we reviewed generic business process models from a variety of published sources (e.g., Davenport (1993), Kotler (1997), Ulrich and Eppinger (1995)). However, the Process Handbook does not force a single perspective on any of these processes. It can store different views of a process as alternative specializations.

8. REFERENCES

Ahmed, Zia. 1998. “An Integrated Dependency Editor for the Process Handbook.” in *Department of Electrical Engineering and Computer Science*. M. Eng. Thesis. Cambridge, MA: Massachusetts Institute of Technology.

AT&T Quality Steering Committee. 1992. “Benchmarking: Focus on World-Class Practices.” . Indianapolis, IN USA: AT&T Bell Laboratories.

Baligh, H.H., R.M. Burton, and B. Opel. 1990. “Devising expert systems in organization theory: The Organizational Consultant.” Pp. 35-57 in *Organization, Management, and Expert Systems*, edited by M. Masuch. Berlin Germany: Walter de Gruyter.

Bernstein, Abraham, Chrysantos Dellarocas, Thomas W. Malone, and John Quimby. 1995. “Software Tools for a Process Handbook.” *IEEE-Data Engineering* 18 (1):41-48.

CIO Magazine. 1992. “Back Support for Benchmarkers.” *CIO Magazine* June 16.

Crowston, Kevin Ghen. 1991. “Towards a Coordination Cookbook: Recipes for Multi-Agent Action.” in *Sloan School of Management*. PhD Thesis. : MIT.

Cusumano, Michael A., and Richard S. Selby. 1995. *Microsoft Secrets: How the World’s Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People*. New York, NY: The Free Press.

Davenport, Thomas. 1993. *Process Innovation: Reengineering Work through Information Technology*. Boston MA USA: Harvard Business School Press.

Gasser, Les. 1992. “HITOP-A: Coordination, Infrastructure and Enterprise Integration.” Pp. 373-378 in *Proceedings of the First International Conference on Enterprise Integration Modeling,* edited by C. J. Petrie Jr. Hilton Head, SC, USA: MIT Press; Cambridge, MA, USA.

Geisler, Martha A. 1995. “The Evolving Healthcare Delivery Systems: Applying the Process Handbook Methodology to Gain a Vision of the Future.” in *Sloan School of Management*. M. S. - Thesis. Cambridge, MA: Massachusetts Institute of Technology.

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

Hammer, Michael, and J. Champy. 1993. *Reengineering the Corporation*. New York: Harper Business.

Harrington, H.J. 1991. *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competetiveness*. New York NY USA: McGraw-Hill.

Herman, G., M. Klein, T.W. Malone, and E. O'Donnel. 1998. “A Template-Based Process Redesign Methodology Based on the Process Handbook.” . Unpublished Discussion Paper. Cambridge MA: Center for Coordination Science, Sloan School of Management, Massachusetts Institute of Technology.

Kaftan, Coleen, and Louis Barnes. 1991. “Sun Hydraulics Corporation.” . Case. Boston, MA: Harvard Business School.

Kettinger, W.J., and V. Grover. 1995. “Toward a Theory of Business Process Change Management.” *Journal of Management Information Science* 12 (1):9-30.

Kettinger, W.J., S. Guha, and J. T. C. Teng. 1995. “The process reengineering life cycle methodology: A case study.” Pp. 211-244 in *Business Process Change: Concepts, Methodologies and Technologies*, edited by V. Grover and W. J. Kettinger: Idea Group.

Kettinger, W.J., J. T. C. Teng, and S. Guha. 1997. “Business Process Change: A Study of Methodologies, Techniques, and Tools.” *Management Information Science Quarterly* 21 (1):55-80.

Kotler, Philip. 1997. *Marketing management : analysis, planning, implementation, and control*. Upper Saddle River, NJ: Prentice Hall.

Kruschwitz, Nina, and Geaorge Roth. 1999. “Inventing the Organization of the 21st Century: Producing Knowledge through Collaboration.” . Sloan Working Paper. Cambridge, MA: MIT-Sloan School of Management.

Leavitt, Wilder. 1995. “Health Care Delivery Systems: Using the MIT CCS Handbook to Create Organizations for the 21st Century.” in *Sloan School of Management*. M. S. - Thesis. Cambridge, MA: Massachusetts Institute of Technology.

Lyon, William. 1995. “The Process Handbook Supply Chain Reengineering.” in *Sloan School of Management*. M. S. - Thesis. Cambridge, MA: Massachusetts Institute of Technology.

Majchrzak, A., and L. Gasser. 1992. “HITOP-A: A tool to facilitate interdisciplinary manufacturing systems design.” *International Journal of Human Factors in Manufacturing* 2 (3):255-276.

Malone, Thomas W., and Kevin Crowston. 1994. “The Interdisciplinary Study of Coordination.” *ACM Computing Surveys* 26 (1):.

Malone, Thomas W., Kevin Crowston, Jintae Lee, Brian Pentland, Chrysanthos Dellarocas, George Wyner, John Quimby, Charley Osborn, Abraham Bernstein, George Herman, Mark Klein, and Elissa O'Donnell. 1999. “Tools for inventing organizations: Toward a handbook of organizational processes.” *Management Science* 45 (3):425-443.

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

Mi, Peiwei, and Walt Scacchi. 1993. “Articulation: An Integrated Approach to the Diagnosis, Replanning and Rescheduling of Software Process Failures.” Pp. 77-84 in *Proceedings of 8th Knowledge-Based Software Engineering Conference*. Chicago, IL, USA: IEEE Comput. Soc. Press; Los Alamitos, CA, USA.

Nau, D.S. 1987. “Automated Process Planning Using Hierarchical Abstraction.” *TI Technical Journal* 39-46.

Ruelas Gossi, Alejandro. 1995. “Inventing Organizations for the 21st Century in Mexico: Supply Chain Management in a Brewery.” in *Sloan School of Management*. M. S. - Thesis. Cambridge, MA: Massachusetts Institute of Technology.

Sacerdoti, E.D. 1974. “Planning In A Hierarchy Of Abstraction Spaces.” *Artificial Intelligence* 5 115-135.

Salancik, Gerald R., and Huseyin Leblebici. 1988. “Variety and Form in Organizing Transactions: A Generative Grammar of Organization.” *Research in the Sociology of Organizations* 6 pp. 1-31.

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

Ulrich, Karl T, and Steven D. Eppinger. 1995. *Product Design and Development*. New York, NY: McGraw-Hill.

Wyner, George M., and Jintae Lee. 1995. “Applying Specialization to Process Models.” Pp. 290-301 in *Conference on Organizational Computing Systems (COOCS)*. : ACM.

Zlotkin, Gilad. 1995. “Coordinating Resource Based Dependencies.” . Working Paper. Cambridge, MA: MIT - Center for Coordination Science.