Implementing Critical Success Factors in software reuse

by M. Wasmund

Software reuse is one of several technologies that can improve quality and effectiveness of software development. The introduction of a reuse infrastructure within an existing organization and the associated modification of employee behavior and processes is a complex interdisciplinary task. The structuring and monitoring of several coordinated activities is required in order to be successful. This paper describes a practical application of the Critical Success Factors method on reuse technology insertion into the software development process. The Critical Success Factors method has proved to be a useful means for the introduction of software reuse concepts. Application of the method and results are discussed in detail, concluding with lessons learned and recommendations for similar efforts.

Most major companies are adopting total quality management techniques in order to achieve productivity and quality goals essential for successful competition in today's marketplace. IBM's total quality management approach is called market-driven quality (MDQ). In the area of software development, several efforts are underway to achieve MDQ goals such as defect prevention, cleanroom techniques, and computer-assisted software engineering (CASE) methods. One major element of MDQ is the demand for dramatically increased reuse of valuable assets such as software, designs, and experiences to prevent redundant development and maintenance efforts. In the late 1980s, IBM launched a worldwide campaign to implement reuse formally into the processes of its internal operations,[1] beginning with the software development process. The effort is spreading beyond the reuse of pure software toward reuse of any valuable assets such as documents, process specifications, or test cases. In this paper, implementing reuse techniques means not only the easier task of modification and documentation of current processes, but includes the essential work to initiate and sustain changed behavior in all people who are part of a defined process. Several coordinated activities are necessary to change processes toward the desired direction. An example for this is described in Reference 2. The effectiveness of the process transformation depends on the appropriate selection of activities and their careful monitoring and control.

At the IBM system software development site in Böblingen, Germany, the first reusable parts center was established in 1987, followed by the formal documentation of the reuse process in 1989. Alternative processes are discussed in Reference 3. The objective of the parts center was, and still is, the production of highly generic reusable software components for worldwide use within IBM. As part of the corporate-wide insertion of the IBM reuse methodology, the author, who is the reuse site coordinator at IBM Böblingen, applied the *Critical Success Factors* method.[4] The method proved to be very helpful for this type of task; it supported assessment of the already established

parts of the reuse infrastructure, and provided a focus on future directions. This is the first known application of Critical Success Factors (CSF) on reuse. Application of the method and the obtained results are displayed step-by-step within this paper.

Results showed us that pure application of the method is not sufficient for effective technology transfer; we underestimated the nontechnical aspects of this task. Recommendations for future efforts are provided in this paper.

We first familiarize the reader with the basics of the CSF method, illustrated by a nontechnical example. The method is then applied to the task of reuse technology insertion. The resulting factors and activities are discussed extensively. We then examine the effectiveness of the method by providing details of some of the activities. The lessons learned and our recommendations for similar efforts conclude the paper.

Principles of the Critical Success Factors method

The Critical Success Factors method basically allows one to create a project out of a problem definition. This is done by decomposing a well-defined goal into a comprehensive list of subgoals, called *factors*. From there a list of *activities* follows, whose purpose is to obtain the factors and eventually accomplish the specified goal. The activities are executed in a project context, leading to the solution of the original problem. Compared to other, more intuitive methods, CSF provides an easy means to keep an overview about all relevant activities necessary to be executed in order to solve a problem. It ensures that no essential activities are omitted. Furthermore, it provides a means to recognize and eliminate redundant activities in order to focus available resources on crucial topics.

CSF has been used for problem solutions for years. John F. Rockart published it at MIT in 1981.[4] Kurt Nagel adopted it to the area of enterprise management.[5] The method is successfully applied to all sorts of problems and taught at business management classes. However, we believe this is the first published case that discusses where CSF was applied to inserting reuse technology into software development.[6] The potential of the method is illuminated by the fact that other organizations within IBM and outside of IBM are beginning to use it for the same purpose as described in this paper.

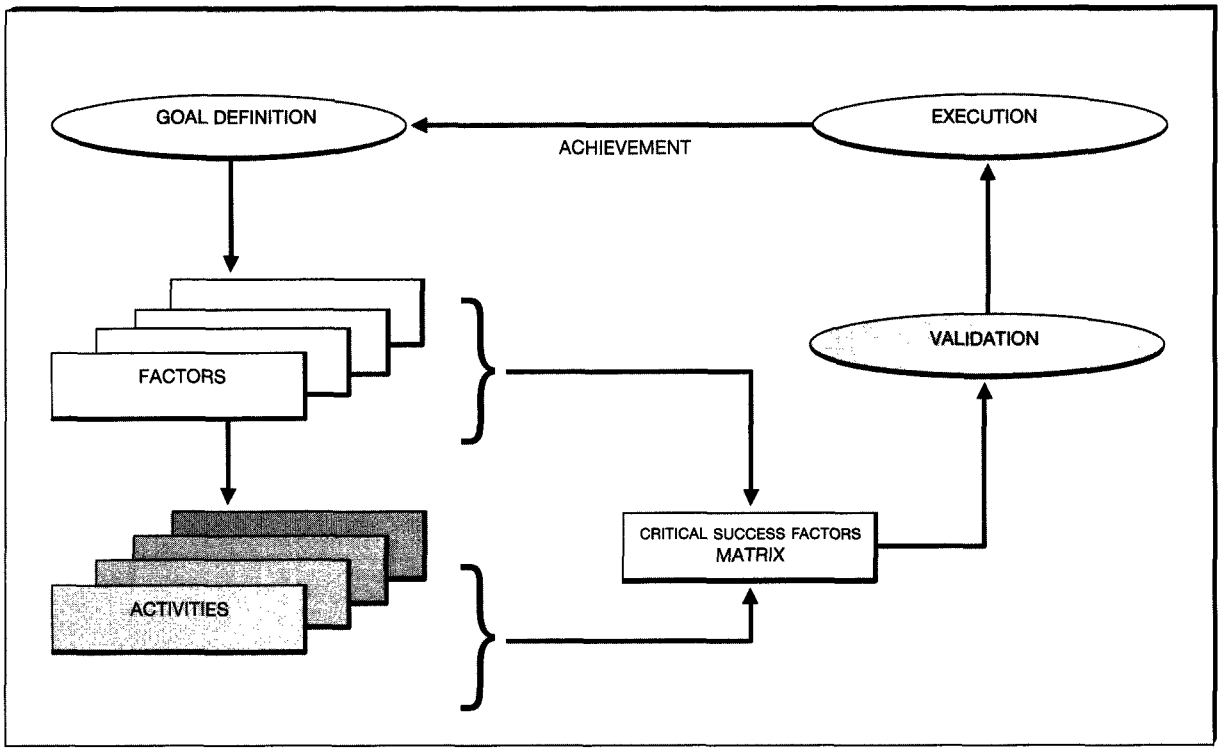
The basic steps of the CSF method are illustrated in Figure 1 and outlined below. A goal to be achieved is decomposed into a set of factors. Activities supporting the factors are then entered into a matrix and their relationship to the factors is validated. Finally, the activities are performed. The steps are:

1. Define the goal. Since the succeeding steps depend heavily on the exact goal definition, statements about purpose, scope, and time constraints must be included in the goal description and be as specific as possible. Let us illustrate this by a nontechnical example:

   Example: Lost on an island. People who are lost on an island may specify as their (imprecise) goal: survival. The critical factors to accomplish the goal depend on the maximum time expected until rescue. If rescue is expected within 24 hours, availability of food may not count as a critical factor, but prevention of hypothermia certainly does. On the other hand, if rescue is expected within 24 days, availability of food is an essential success factor.

   It is essential to be as specific as possible, including the exact definition of quantities. Slight modification of the goal may lead to major changes in the required set of factors and derived activities.
2. Decompose the goal into a set of factors. This step should not say anything about implementation, and therefore the factors do not contain any verbs. The factors describe things or entities that must be obtained in order to reach the goal. The optimum is a disjunct set of factors, being as independent or orthogonal from each other as possible. This step is not finished until the distinction between essential and "nice-to-have" factors is defined, leading to a minimized set and preventing creation of redundant activities in Step 3. The correct set of factors is achieved if the omission of only one factor makes it impossible to accomplish the goal.
3. Define activities. In contrast to Step 2, activities should always contain verbs to express the work to be performed, to satisfy one or

Figure 1 Steps of the Critical Success Factors method



several factors. Continuing the example of Step 1, looking for a stream may be one of several activities to satisfy a critical success factor called water.

4. Build and validate the CSF matrix. Factors and supporting activities are entered into a correspondence matrix, showing which activities support which factors (see Table 1). The ties between factors and activities are marked. The obtained matrix can be used for several purposes. First, it allows recognition of unsupported factors, i.e., factors for which no activity has so far been defined. Second, it allows elimination of redundant activities. In Table 1, activity 6 supports the same factors as activity 7. Activity 6 is therefore redundant and can be eliminated, thus allowing focus of available resources on crucial topics. Third, the CSF matrix may serve as a project management ingredient to track and readjust activities, making their relationships to the success of the overall goal visible.

Table 1 Generic example of a CSF matrix

Critical Success Factors						Activity
A	B	C	D	E	F	
X	X			X	X	1
X	X	X			X	2
			X			3
X						4
	X		X			5
X			X	X		6
X		X	X	X		7

5. Execute the activities. This step may sound trivial, but our experience shows that this is a nontrivial task and uncovers all kinds of problems. This will be further evaluated in a later

section that discusses the experience of applying this method to reuse technology insertion.

As in software development, reiteration of these sequences of steps may be necessary due to changed goals or environment.

Application of the CSF method on reuse technology insertion

Having introduced the principles of the Critical Success Factors method, its application to the internal operations within an IBM software development site at Böblingen are next described. The discussion is enriched by on-site experiences, so that the reader may learn from these experiences and "reuse" them.

Step 1: Definition of the goal. As stated above, the goal should contain purpose, scope, and timing. Our goal was: Establish a well-defined reuse technology insertion program leading within two years to (1) a shortened development cycle, (2) increased reliability of marketed products, and (3) highest reuse maturity[7] within IBM. The rationale of the goal is straightforward: In order to serve customers more rapidly with solutions to problems and to obtain their optimum satisfaction with our products, it is necessary to build the software products faster and error-free.

Step 2: Determination of critical success factors. This step defines the factors that are required to accomplish the stated goal. There seems to be no structured way to obtain these factors, but we learned that brainstorming and intense discussions with knowledgeable people may be the best way. S. D. Fraser applied a group decision support system to reuse[8] based on Interpretive Structural Modeling.[9] There is a temptation to focus too early on activities rather than factors. A principle of software development thus applies here as well: The greater the effort spent in Steps 1 and 2 of this method to obtain accurate and agreeable statements, the less rework is required to achieve the overall goal.

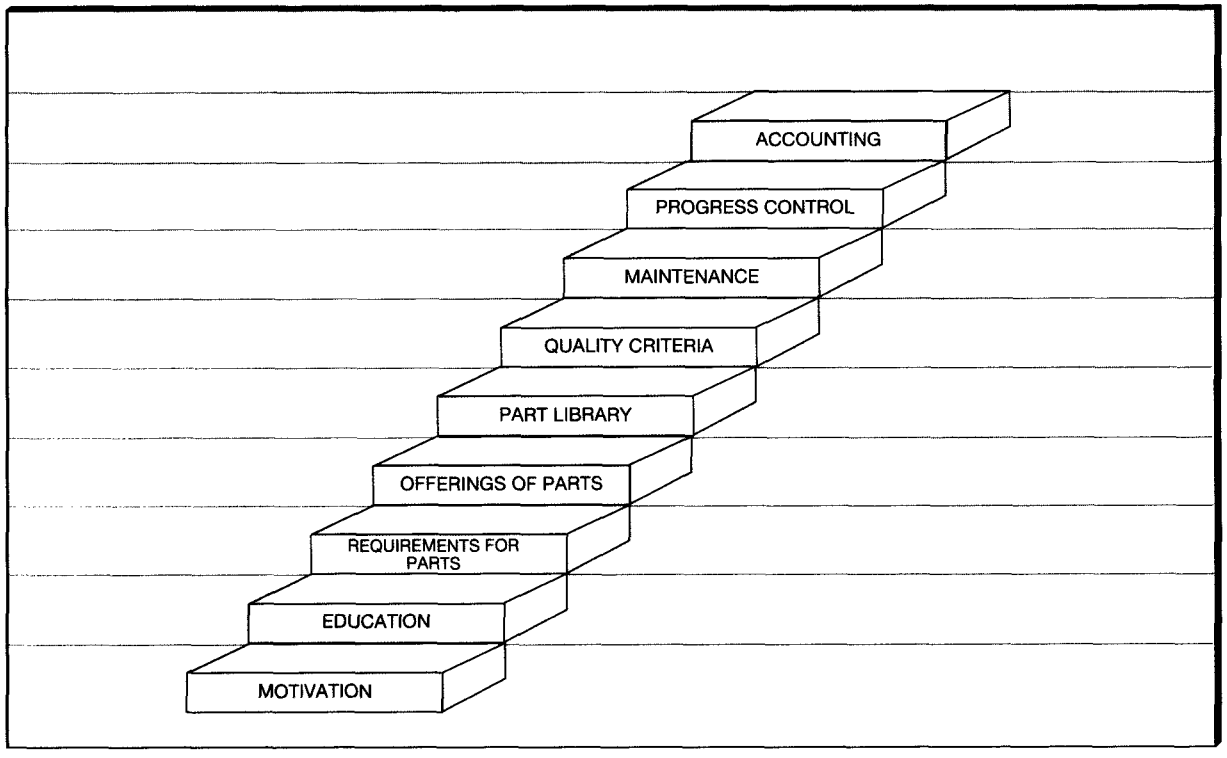
Analyzing the goal stated above for our organization, we found it to have typical attributes of *trading*. In order to enable software and associated assets (design descriptions, test cases, document templates) to have multiple application to different environments, it is necessary to establish a trading infrastructure or reuse marketplace to link customers and suppliers. The elements of a marketplace are derived from the fact that suppliers offer parts and customers require parts. Therefore, information about offerings and requirements is necessary. To store and advertise the parts to be traded, a repository for holding the parts as well as their description is needed. The probability of a part exchange also depends heavily on the trust in or quality of the parts. Because quality is a rather unclear word, we introduce the term *certification level* here, which means a well-defined guaranteed completeness and defect rate level of the offered parts. Closely related to this is the issue of maintenance, for two reasons. First, the potential parts user wants to have a clear view of the dependencies that the overall product has on others. A product-lifetime guarantee for the included part is normally expected. Second, if a part is repetitively used in different contexts, the economical benefits are best when there is only one centrally maintained version of the part.

As in any trading environment, some kind of accounting is required in order to record exchanges and associated costs and savings, and to measure the effectiveness of the trading infrastructure. A technology insertion approach such as reuse requires motivation of all involved, namely the professionals who have to change their practice, and the management who control the resources. The management particularly is very interested in controlling the progress toward the specified goal, so this is to be included into the list of critical success factors. Progress control, another factor, includes the definition of progress, checkpoints, and milestones. Both management and professionals need education to create the appropriate mind-set and establish the technical skills to practice reuse effectively.

The result of this step is a disjunct set of factors as depicted in Figure 2. Note that the application of the CSF method to a different environment will result in different sets of factors. Teaching this method in lectures showed a surprising variety of CSF sets created by the students out of slightly varying goal definitions. An alternate set of factors is discussed in the later section on experiences.

Step 3: Definition of required activities. Next, we briefly outline what activities we found when applying the CSF method at our site. Detailed de-

Figure 2 Critical Success Factors enabling the objective defined in Step 1



scriptions of activities, experiences with them, and some lessons learned are included in Step 4 and in the next section.

Again, there is no strict rule for deriving activities satisfying the set of CSFs obtained in the previous step. Brainstorming within a group of experts and practitioners gave us good results. At this stage, it is essential to consult everyone who is involved in the environment to be changed or who might be affected by the changes, otherwise the execution of the defined activities will meet with low acceptance.[10]

Motivation. The first basic activity that comes to mind when looking for a means to increase motivation is information about the benefits of reuse for the company and the individuals through appropriate communication channels. A commonly used means to motivate people are incentives, particularly in the early stage, when the maturity of the organization in terms of reuse deployment is still low. Depending on the culture of the organization, the effectiveness of the associated activity, run incentive program, varies. In a later stage, when reuse becomes a more common practice (maturity level > 2),[7] checkpoints within a modified development process remind and support the professionals. The return on investment, if positive, will motivate an organization to make more expanded use of this method. Quantitative objectives for reuse are a particularly strong motivator if the objective is realistic and achievable.

Progress control. This is not possible without standardized measurements. The purpose of progress control is not only to show successful cases of technology insertion to others, but also to assess the contribution of reuse to the overall goals of the product within a modified development process. For automation of reuse progress reports, tool support is needed.

Part library. A part library or repository can initially be a simple list. In order to make it publicly known and accessible, communication channels

are to be established. As the content of the list of parts grows, structured database search mechanisms are to be offered by supporting tools. For meaningfulness and comparison of the library contents, classification standards are required.[2] To start off a successful reuse program, a critical mass of reusable parts must exist, since no one will look into a scarcely populated repository.

Offerings of parts. Offerings of parts are solicited through incentive programs, appropriate communication channels, and by locating additional sources for parts.

Requirements. Requirements for parts can be solicited through similar activities. Additionally, specific checkpoints at the early stages of a modified development process (i.e., in case of the waterfall model, during the requirement and design phase) yield a list of required reusable parts for a particular product.

Quality criteria. These guarantee a certain level of trust in the available parts. Approved and commonly applied standards are needed to establish quality levels for reusable parts. At IBM, this is called certification levels as depicted in Table 2. Appropriate modifications of the development process foster the application of quality criteria to reusable parts in order to avoid dissatisfaction of users, who may have integrated unreliable components into their product. In the table, the lowest level "as-is" at the bottom of the table designates software that was not originally designed to be reusable, but which is of potential use to others. The medium level "complete" designates a piece that can be reused without additional explanation and which enjoys support on an availability basis. The maximum level "certified" is desirable to prevent multiple maintenance and designates a very high level of trust into that piece.

Maintenance. As for any software, reusable software is to be maintained throughout the life of all dependent products. This means that the life cycle of a reusable part is normally longer than of a specific product. To reflect the cross-product attributes of reusable part maintenance, the development process must be updated. Supporting tools are required to forward any corrections or updates to the dependent products and to store the different versions of a reusable part in a product-independent configuration management library.

Education. Education about reuse can be accomplished through oral and written presentations, classes, workshops, and symposiums. It is best done by establishment of an appropriate curriculum addressing all reuse-related issues, e.g., designing for reuse, integration of reusable parts, availability of parts, and calculation of the business case. The curriculum at IBM is tailored to the target audience. A well-staged sequence of courses is offered to professionals; briefings about reuse address the interest of the management team.

Accounting. Accounting of reuse-related cost savings, usage statistics, and parts offered needs: (1) standardization of how and what to account, (2) enhancement of the development process to define when to account, and (3) supporting tools to hold and calculate accounting data.

Step 4: Correspondence matrix and validation. Following the CSF method, we are at the bottom of Figure 1. The residual steps now direct us to reverse the direction to validate the correctness of our results.

Having defined all the activities that we believe are necessary, it is now time to obtain an overall picture. The matrix shown as a template example in Table 1 is populated with the set of factors obtained in Step 2, and with the set of activities obtained in Step 3. The result is a two-dimensional array of factors and activities. The core of the matrix is filled with junction operators (marked by an X) as depicted in Table 3. We next discuss the rationale for the position of the X when validating the activities. When executing this step, one may observe that a particular activity supports more success factors than expected; other activities may be rendered redundant during this step.

The amount of intersections per row helps to derive priorities; however, this can be misleading. One should not derive priorities from the amount of X alone, but also from careful evaluation of each activity and its current state. As an example, "find sources for parts" relates only to one CSF in our matrix; nevertheless, it is essential to achieve the overall goal. Because each of the activities is essential, we associate the numbering of the activities in Table 3 in this context with the resources spent to achieve the desired goal.

The following discussion of derived activities contains a description of the checking performed

Table 2 Certification levels of reusable assets

Quality Level	Maintenance Support	Test Status	Documentation
Certified	Full	Independently tested	Comprehensive
Complete	Limited	Tested by originator	Complete
As-is or accepted	None	Uncertain	Incomplete

Table 3 Interrelationship of Critical Success Factors and supporting activities

Critical Success Factors									Activity
Motivation	Progress Control	Part Library	Offerings of Parts	Requirements for Parts	Quality Criteria	Maintenance	Education	Accounting	
X	X			X	X	X		X	1. Modify development process
X			X	X					2. Run incentive program
X		X	X	X					3. Establish communication channels
X	X	X			X			X	4. Define and apply standards, including measurements
		X							5. Find sources for parts
X							X		6. Establish curriculum
	X	X				X		X	7. Have supporting tools

to show how the activities support related critical success factors. Even more important, it gives the opportunity to share with the reader our experience of executing these activities.

Activity 1: Modify development process. Reviewing and reshaping the traditional software development process, i.e., the way software is built today, helped us in gaining most of the defined critical success factors. This work is linked to another activity not directly related to reuse technology insertion. ISO 9000,[11] the international standard addressing quality, expects an organization to describe all its processes in a formal way, preceded by a review of all important processes. This facilitated the necessary updates required by the introduction of reuse to a great extent.

The following is a summary of the major process updates, starting from the classical waterfall model. The first phase, where the requirements of a future product are discussed, has already been subjected to changes. We now subdivide the estimated amount of code to be shipped into different categories. The most important categories for the purpose of this paper are the amount of *product-unique* code and the amount of *reused* code. Among other benefits for project management, this separation eventually leads to a list of requirements for parts. Putting the quantities of the different code categories into the product plan

supports progress control. The state of reuse is collected in every development step. Some of the questions that should be raised in a modified development process in order to maximize the benefits from reuse are: how much code is expected to be taken from or to be owned by others, how much code is expected to be provided to others or a product-specific reuse library, have the projections been met, and finally, if not, for what reasons.

Very closely linked to these considerations is the establishment of a return-on-investment case for the project in terms of reuse. This gives project managers a better understanding of the benefits of employing reuse methodology and may be the final clue to convincing hesitant people to go new ways.

As mentioned in Step 2, the quality of reusable parts is even more crucial than that of product-unique parts. In order to maximize the number of usages of a particular part, its quality must be rigidly checked before handing it to others, and its quality level must be communicated together with the other attributes of the part to the development community in a way everyone understands. For this purpose, IBM established three distinctive levels of quality called *certification levels*. Table 2 shows the three levels that were previously discussed.

Two major obstacles for reuse insertion remain to be discussed, maintenance and accounting. Uncertain maintenance restricts usage of a part to internal tools since market products cannot rely on uncertain maintenance of imbedded software parts. Maintenance effectiveness for reused parts is gained by redefining the appropriate process step such that error reports of customers or field engineers can be quickly routed to the owner of an erroneous component. The underlying assumption is that more and more products are composed of building blocks owned by different organizations rather than created completely new each time one is required.

The multiple use of one part in different products accelerates the maturation of the part toward the zero defect point, because more users in different environments will uncover hidden defects. On the other hand, one defect of such a part hits several different products. Therefore, the initial defect rate of a reusable part is required to be much lower than usual.

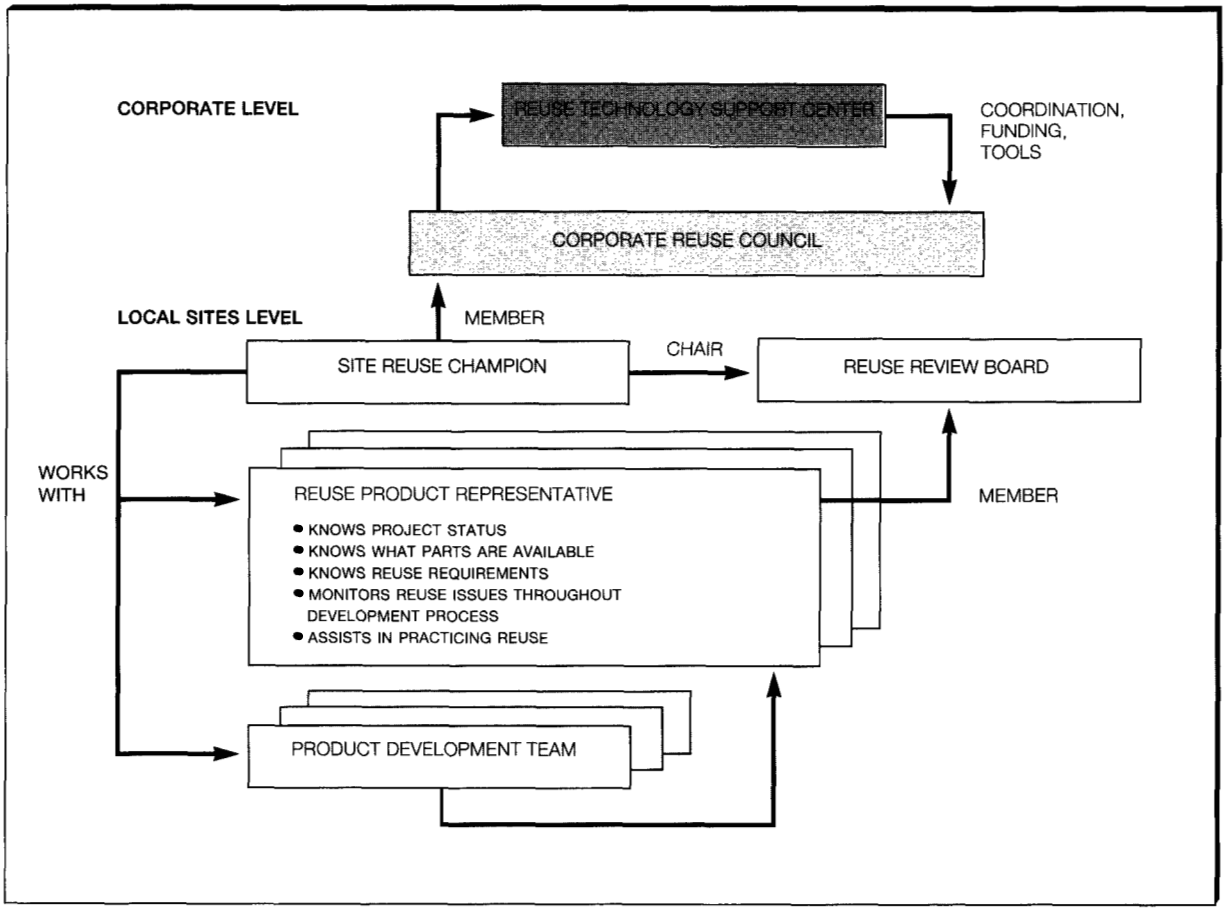
To exploit the economic attributes of reuse to its full extent, new accounting and charging methods, including internal standards, are needed. This would also satisfy the rapidly increasing demand to charge for reusable software. Our site example teaches us that there is a lack of progress determining the value of a reusable part and of availability of flexible charging mechanisms between organizations.

Our overall experience with software development process modifications is that, first, a minimum maturity of the unmodified development process is required. In terms of maturity levels,[7,12] which range from 1 (lowest) to 5 (highest), a level of 3 is a good basis with which to start. Less mature processes can also be changed, but the effort is much greater compared to the obtained results. Second, it is relatively easy to change process documentations, but extremely hard to insist on the actual day-to-day execution of the process according to the new documentation. This obstacle can be partially removed by providing skeletons for all project-relevant documents, which have entries for reuse issues. When preparing project status reports, etc., users of the skeleton are reminded to document reuse-related issues.

Activity 2: Run incentive program. The outlined development process changes are effective, but take time. Technology insertion is a long-term enterprise, but our specified goal states a time frame of two years, so we need an accelerator. Incentives are commonly used to change human behavior or boost performance. IBM adopted incentive programs for reuse in all major software development organizations. Most of these incentive programs are point based, such as that used by the airline industry in a frequent flyer program. The provider of a reusable part gets credits depending on the size and usage of the part, and the user gets credits for integrating available parts instead of writing specialized software. Monetary or nonmonetary awards are paid when a certain point level is reached. IBM Böblingen created a reuse quality award program to speed up the desired transformation of the development process. The attributes of this program follow.

- Rewards for including parts in a product as well as providing parts to others
- Consistency with already existing award pro-

Figure 3 Infrastructure supporting reuse at IBM



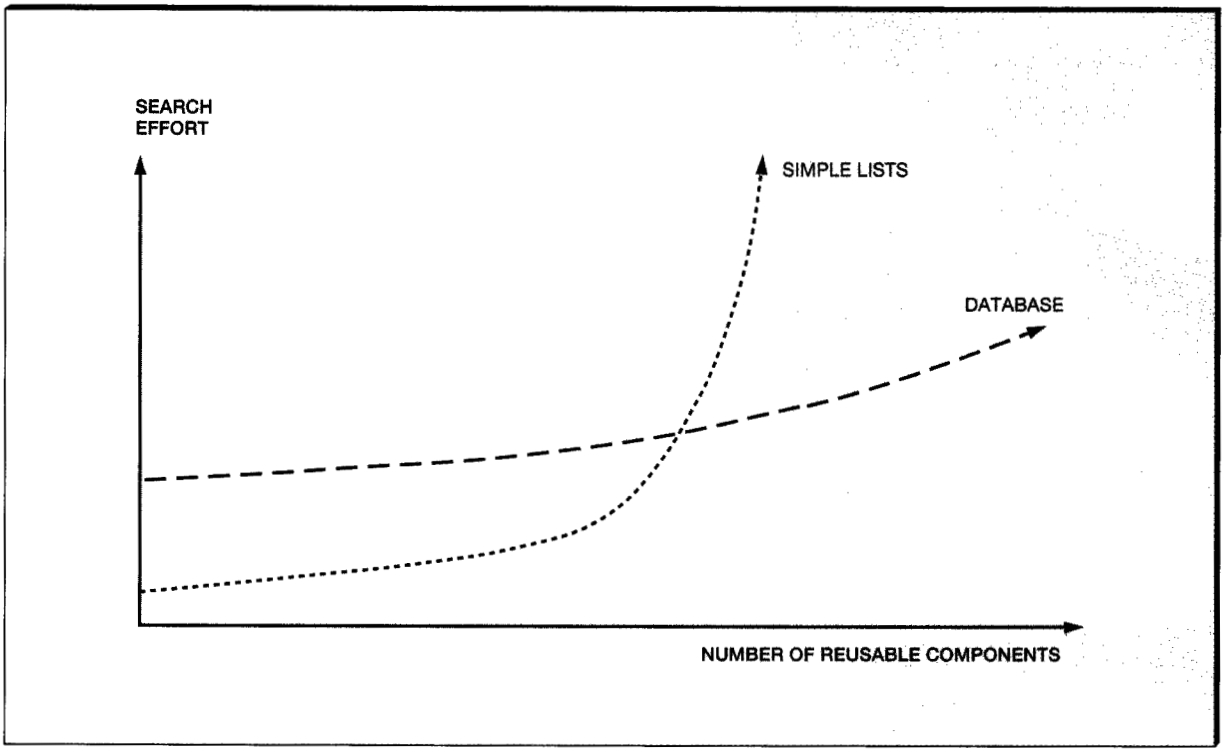
grams (there is a quarterly contest, rewarding quality-related improvements)
- Consistency with and usage of established reuse measurements, instead of establishing a sophisticated point-based scheme
- Rewards for usage of unmodified code only (otherwise multiple maintenance of a part weakens the economic benefits)

This incentive program aims to increase motivation, quantity of parts offered, and requirements for parts. At this time, the incentive program is too new to reveal experiences; however, considering the economic benefits of reuse, incentives are a very inexpensive investment.

Activity 3: Establish communication paths. It is straightforward that the trading of reusable parts depends heavily on availability and effectiveness of communication channels between suppliers and customers. We subdivided this activity into the establishment of four communication channels.

Personal communication always proves to be the most important channel when working with representatives of product areas. The representatives serve as a bridge between product areas for exchange of any information relevant to reuse. Figure 3 shows the responsibilities of these people at IBM. Their effectiveness depends on the selected person, who must be recognized as a competent professional. Each site has a board consisting of representatives from all product development areas. The board coordinates reuse-related activities within the site and is connected

Figure 4 Comparison of search efforts for different storage media



to other sites through the site "champion" who is also a member of the Corporate Reuse Council.[1,13]

IBM conducted an internal analysis of media used for communication between professionals, and determined that electronic bulletin boards are the most favored medium. Thus they are broadly used for exchanging experience between sites, asking for reusable parts, and offering help in practicing reuse techniques.

During the beginning of reuse deployment, the relatively small amount of reusable parts still allows a list of parts to be useful when disseminated. At our site, we gather information about reusable parts from all sources within the corporation and provide a consolidated sorted list of these parts to the software development engineers.

As a fourth communication channel, IBM developed a sophisticated database for corporate-wide storage and retrieval of reusable parts for internal use. It is similar to literature databases, which can be considered as a reuse database of documented experiences. The database supports and enforces a set of standards for reusable parts. It also supports forwarding of maintenance notes to subscribed users of reusable parts. Figure 4 shows the effectiveness of communication channels depending on the search space. Searching simple lists for reusable parts works very well in the range of a maximum of 500 parts. Well-known mechanisms, e.g., string search in editing mode, prevent the user from learning new tools. Larger amounts of parts require sophisticated database search mechanisms in order to maximize the probability of hitting the needed part. Our experience shows that a large-scale repository needs to be complemented by simple lists of available parts.

Summarizing Activity 3, we see that it supports four critical success factors (repository, offerings, requirements, and maintenance). In order of priority, the correct selection of product repre-

sentatives and easy-to-use parts list are the most important subactivities during an initial reuse effort.

Activity 4: Define and apply standards and measurements. All technology insertion efforts covering multiple environments require standards in order to (1) track and compare progress, (2) have a base for business calculations, (3) show consistent cost-savings diagrams, and (4) have reliable and commonly accepted quality criteria for reusable parts. Experience shows that covering multiple sites or organizations by the same standards requires careful cooperative definition by a central body. IBM established the Reuse Technology Support Center[13] for this purpose. Their work resulted in a comprehensive set of guidelines and standards. The basic and most important ones are measurements for the number of reused lines of code and for the lines of code provided to others. Financial and business case calculations, as well as productivity figures, can be derived from these.

Validating the relation of activity 4 to the critical success factors, a number of factors are supported by this activity. The motivation to utilize reuse techniques is higher if the value of these techniques can be assessed by usage of standardized measurements. Measurements also allow quantitative progress control as an ingredient of project management. The value of parts offered is greatly enhanced if the user can rely on part descriptions based on agreed upon and well understood measurements. In addition, searching for parts is much easier, if there is a classification standard. IBM, therefore, has introduced such standards. Quality criteria and certification levels of available parts are to be based on approved standards; they are essential to guarantee the potential customers a certain level of trust, avoiding unsatisfactory experiences. A dissemination of bad experiences would weaken the deployment of reuse technology seriously. Last but not least, accounting is difficult if there are no defined measurements.

Activity 5: Find sources for parts. As Will Tracz[14,15] emphasizes, before we can practice reuse, there must be parts to be reused. Thus to begin with, a base or critical mass of reusable parts must be available from the beginning. At IBM Böblingen, we were in the privileged position to not have to start from scratch, but to have several sources for reusable parts already available. We found and used four distinct sources of reusable parts.

As mentioned earlier, electronic bulletin boards are a prime source for both information and parts within our company. This traditional means of communication contained reusable software long before reuse was promoted as a formal program. The difference between this medium and other media especially created for reusable parts distribution is that electronic bulletin boards and conference disks do not yet use the established classification and certification standards. It is more a "use as-is" approach, such as the well-known public domain software repositories on the Internet and other public networks. Despite the varying level of trust in this source and possible legal restrictions on its usage, this source is well appreciated by the development teams, because of its easy and well-known access methods.

Other sites within the company may offer solutions to local problems. For example, in the scientific world, a question can be asked about whether it is quicker to solve an already solved problem once again or whether it is quicker to find the location of the solution. We found that the traditional behavior of a software development organization is to re-invent solutions. This happens for three reasons: (1) to avoid being dependent on the solution provider, (2) because it is considered to be "more fun" to solve the problem locally, and (3) meeting a tight deadline is usually easier to obtain by providing a specialized solution rather than a general solution on a higher abstraction level. For these reasons, sources of parts at other sites were not evaluated to their full extent in the past.

Work is underway to remove inhibitions to use reuse and the early visible results are the intensive sharing of reusable software across organizational and geographic boundaries. This allowed IBM Böblingen to triple its reuse rate within 12 months. However, increasing financial and administrative independence of company divisions poses additional obstacles to widespread sharing of parts, solutions, and experiences.

Another source cited in the literature[16] is the concept of parts centers creating reusable software on demand. This concept allows production of highly generic, encapsulated software building

blocks. A product could be composed of these instead of being a specialized and hard-to-modify piece of software. An important requirement of that source is quick responsiveness to part requirements and close synchronization with parts customers at a very early stage. IBM Böblingen has run such a parts center for years.[17] Different models of parts centers are discussed in a later section of this paper.

The fourth source of reusable parts discussed here is product spin-off. Figure 5 shows Böblingen's reuse process. A development organization or team offers potentially reusable software spin-off to the reuse review board. Depending on the assessed reusability of the offering and on the requirements of other development teams, the review board classifies and qualifies the offered part and provides public access to it in a database. This structure is complemented by one or more parts centers, whose mission is to satisfy requirements common to multiple projects. Production and consumption of reusable parts is stimulated through incentives. Böblingen's model combines two of the discussed parts sources, namely the parts center and product spin-off. The review board serves as a filter mechanism ensuring a minimum quality of the provided parts. Incentives stimulate the cooperation of providers and requesters of reusable parts.

When designing a new product, software that cannot be obtained from other sources should at least be made as general as possible to be of value beyond the actual project. This improves the overall productivity of the development team and also contributes an improved design style toward encapsulation and object orientation. The additional development effort pays off when the software built is reused at least twice.[18,19]

Activity 6: Establish curriculum. In order to get awareness and education, a set of courses was designed to teach development teams how to integrate reusable software into their products, and how to design their products such that the amount of product-specific software is minimized and the amount of software available for other products is maximized. The class work is complemented by computer-based training and comprehensive online information. Experience shows that the classes must be tailored to the audience (such as programmers, designers, or managers) and to the production environment (such as third-generation languages, fourth-generation languages, or assemblers). The concept of reuse is best introduced in the course of transition from third-generation languages to object-oriented (OO) technologies, because these inherently support aspects of reuse.

Activity 7: Have supporting tools. As soon as the deployment of reuse throughout the organization reaches a significant scale marked by a maturity level greater than two, tools support is essential in order to proceed in an efficient mode. We identified three particularly important areas for tools:

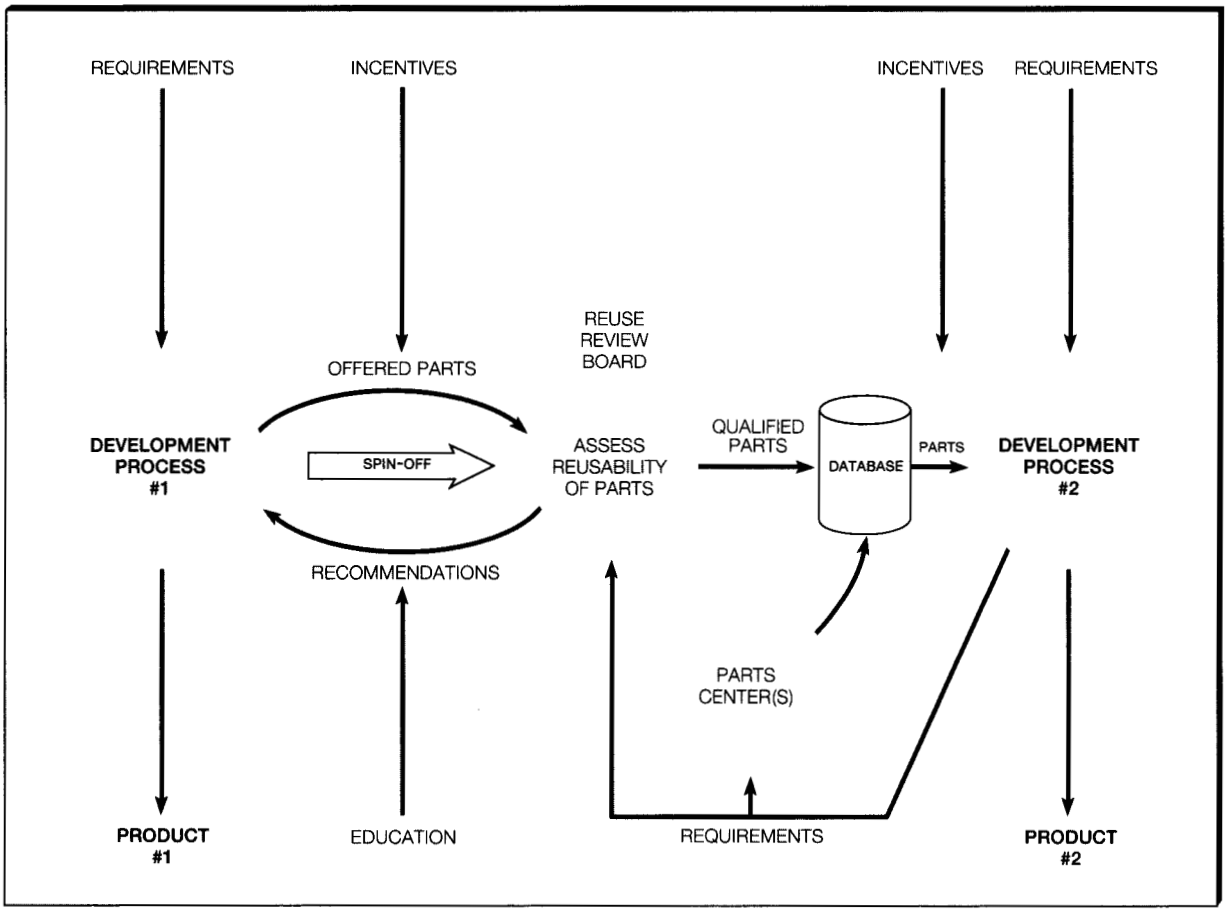
1. A repository (or a list of parts) for reusable parts. This can be a database with a search engine attached to it. IBM developed for its own use such a repository that supports text search as well as faceted search.[1,13] It also provides an automatic update service to subscribed users, in order to distribute news or corrections regarding particular parts.
2. A source code configuration management tool is required to support integration of reusable parts into products, including version control.
3. A code counting tool able to count reused source instructions is required to support accounting and measurement.

Ideally, these three tools can be integrated into a common development (or CASE) environment that can be tailored to all organizations wishing to participate in reuse; however, presently these three tools are not fully integrated, which is a major inhibitor for their widespread use. Early availability of these tools is also very important. Late availability will weaken the progress of reuse because of the handling effort in dealing with reusable parts. Table 3 shows that this particular activity helps to achieve four out of nine critical success factors.

Step 5: Execution of activities. Our experiences and the lessons learned with implementation and execution of the activities are next discussed.

Scope of the method. The application of the Critical Success Factors method helped us to determine, coordinate and track the crucial activities that are required for a technology insertion project such as reuse. The method does not support the implementation and execution of the activities. Implementation has to be carefully tailored

Figure 5 Linkage of development processes by reuse infrastructure

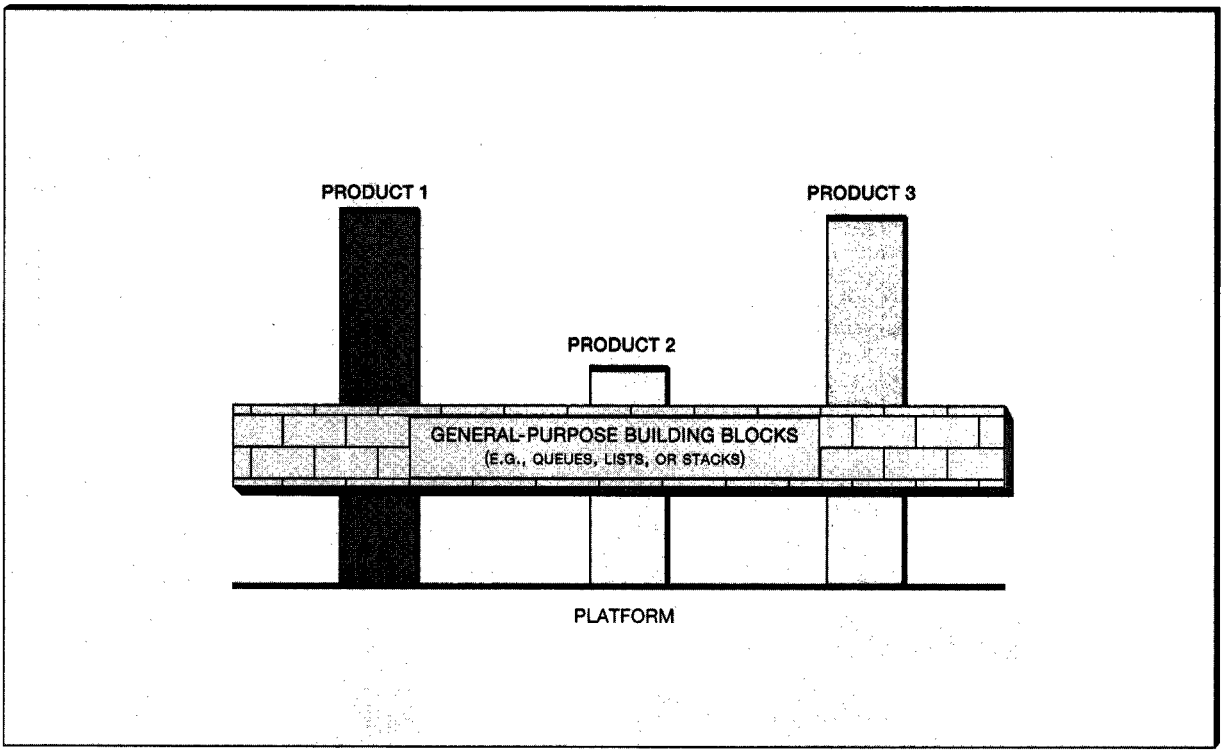


to the local environment and culture in order to be effective.

Edicts versus grass-roots approach. There are two opposite extreme modes of inserting new technologies, grass-roots approaches or edicts. Both ways were applied within IBM as well as within other companies. Edicts tend to quickly generate data to show results. However, evaluation of some cases showed that a crucial ingredient of technology transfer, acceptance, is not necessarily obtained by this top-down approach. On the other hand, pilot projects, as an outcome of grass-roots activities, are only successful if the focus is carefully determined and maintained throughout the project. Continuous management support is required for a lengthy time period. If the pilot project is successful, it serves as a positive example to others. This snowball approach shows the best sustained results in practice. However, it requires careful initiation and a fair amount of patience on behalf of the source issuing the funds to support a project.

Tools versus methodologies. There is a trend to consider introduction of new tools as the quick solution to immature methodologies. Nevertheless, once a methodology is agreed upon and basically accepted, tool support is needed to reduce the effort of the people who want to practice the new methods. For this reason, IBM simultaneously developed a methodology and appropriate tools to allow teams to begin immediately in an effective manner. However, there is still a lack of tool integration. The

Figure 6 Horizontal reuse through general-purpose building blocks usable in almost any product



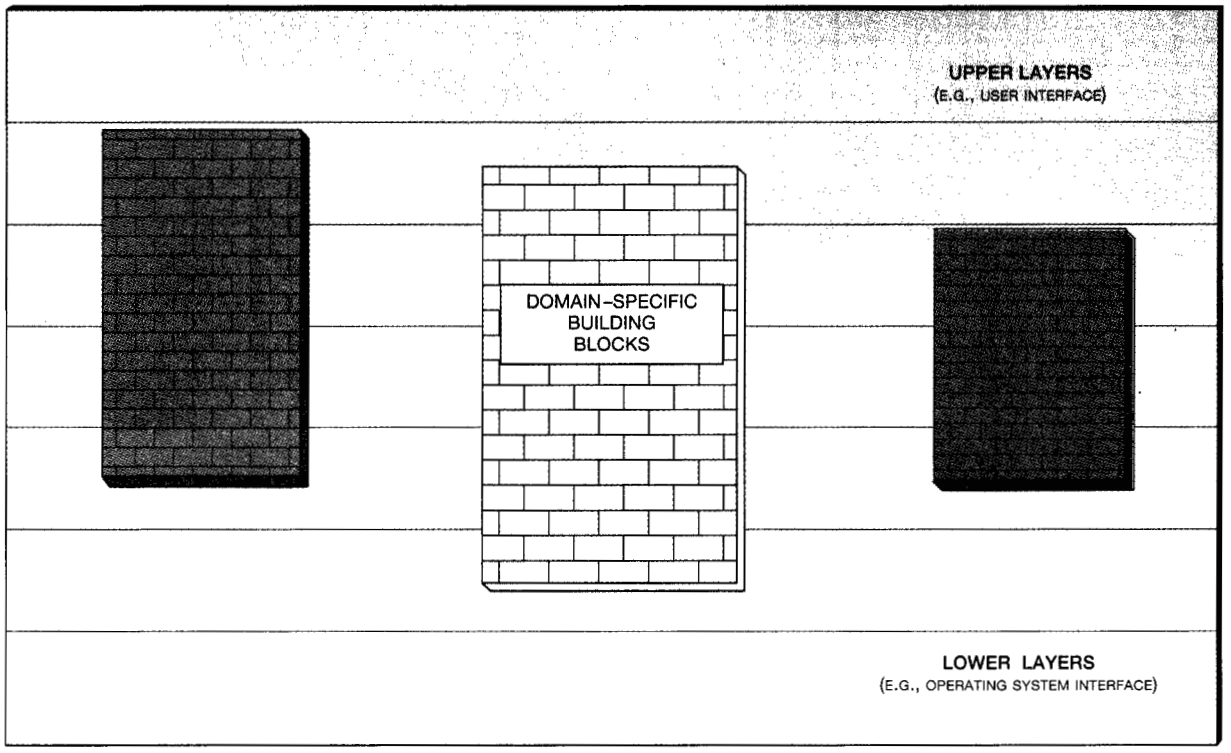
repository containing reusable assets must interact with configuration libraries used throughout the product development, requiring data transfer standards within the company. Establishment of these standards is particularly difficult because of the diverse development environments. This leads to dissemination of very well-designed but independent tools, a common problem in the CASE arena throughout the software industry.

Maturity base. Introduction of a formal reuse technology deals with process changes. In order to be changed, there must be a process to be changed. A minimum process maturity for the development organization must be present to effectively introduce reuse. Ted Davis[12] discussed several maturity models at the Fifth Annual Workshop on Software Reuse. It is also possible to change immature development processes toward reuse or even establish entirely new processes. However, our experience shows that the most effective way is to focus on those areas where well-defined processes are carried out, and to build upon them.

Incentives. Our experience shows that incentives are an inexpensive but effective means to promote new techniques. However, their effect is limited. They help to uncover unknown performance potential of some people or teams, but we do not know of any case where incentives caused an avalanche of interest in new methods. Only those people who already have some affinity to new methods are motivated by incentives.

Parts centers. Comparison of our efforts with other organizations inside and outside IBM, as well as the study of available literature,[10,16] shows that the parts center concept can be successful for a number of reasons. The typical constraints of development organizations, i.e., to deliver products to the market in the shortest time possible, do not usually allow room for additional efforts to produce generalized software. The production of product "building blocks" is therefore to be separated into parts centers. Most successful experiments with reuse rely on this approach. The question is how tightly the parts center should be

Figure 7 Vertical reuse by provision of specialized building blocks tailored to one specific product line



linked to the product development and what the scope of the produced building blocks is. There are two orthogonal alternatives. The first is horizontal reuse through provision of general-purpose building blocks usable in almost any product (e.g., queues, lists, or stacks). This approach is performed by IBM Böblingen's parts center; it was the first center of this type. The second is vertical reuse by provision of specialized building blocks only usable within one specific product line (e.g., client-server communication) across different layers. The topic is extensively discussed in the context of domain analysis.[20] IBM's Federal Systems Corporation in Rockville, Maryland, demonstrates a successful example of this approach.[21] Figures 6 and 7 visualize horizontal and vertical reuse.

Experience shows that the vertical approach is only successful if the product development organization is committed to building parts of higher quality and higher abstraction than needed for the next product release. This means in practice the establishment of a product-specific parts center having a midterm scope with some independence of short-term pressures. There have also been experiments to scavenge code from existing products in order to extract useful code segments for a reuse library; the effort of these experiments has not yet been very convincing, unless there are efficient re-engineering tools available.

Consultation. To learn and practice technology transfer within the scope of reuse, we went through basically two steps. The first of these was teaching and distribution of information about reuse methodology and available parts. The result of this initial step was increased awareness of benefits from reuse methodology, but not actually the application of new methods, because most of the people addressed did not know how to apply the new methods to their specific environment. This forced us to add a second step, advice and consultation. It is evident that the decision to pick up reusable parts for product development happens during the design phase.Therefore, we of-

Table 4 Educational example: A different set of Critical Success Factors and supporting activities, to be completed by the students

Critical Success Factors									Sample Activity
Awareness	Planning	Culture	Involvement	Development Process	Libraries	Tools	Measurements	Education	
									Announce reuse initiative
									Hold kickoff meeting
									Select a site champion
									Select project reuse leaders
									Offer education
									Offer reuse symposium
									(to be filled in)

fered consultation and advisory services to any project that is in the design stage. Experience showed that this offer was welcomed, but not extensively used. One reason for this is that consultation from outside reuse experts was considered as an interference and seen as resulting in dependency. A further reason is that in many projects, design happens very informally and interactively. This makes it difficult to conduct consultation at the correct time. Other sites had similar experiences. We added a third step to our strategy, "fingertip reuse," where a tool is needed that allows the designer to look for reusable parts within seconds, just when it comes to mind. The threshold of asking for assistance or consultation, or dealing with unfriendly tool user interfaces or unsatisfactory tool performance, proved to be too high. A combination of all three steps looks most promising.

Education. As mentioned earlier in this paper, the outlined method served well for teaching classes about reuse process and infrastructure. As Jesus Tirso reports,[13] the set of factors illustrated in Table 4 was created within a class, starting from a similar objective such as described at the beginning of this paper. The idea is to provide to students some of the factors that contribute to improving an organization's maturity and some example activities they might want to consider. There is plenty of blank space for more activities. The students are asked to relate the activities to the CSFs. Along with this matrix, a roll-out plan can be developed for each activity, as well as identification of critical management sponsors, potential resistance or problem areas, and a list of actions to work on these. The education example shows that the concept is very important to the success of reuse. It has already been included in IBM's corporate reuse methodology.

Conclusion

Increased and formalized reuse of software and related assets is one essential method to allow faster delivery of high-quality products to the market. IBM Böblingen initiated a major effort to introduce the concept of reuse into its organization. To find the crucial activities, we applied the Critical Success Factors method. The method and its application are described and evaluated in this paper, as well as the derived activities, all of which prove to be extremely helpful to keep the focus on the correct issues. Execution of the activities and related conditions are discussed, along with a summary of the lessons learned and

recommendations for reuse technology insertion efforts. We conclude that for initiation of such an effort, the Critical Success Factors method is helpful, but the execution of the activities requires iterative readjustment to local conditions and experiences.

Cited references and note

1. J. R. Tirso, "Establishing a Software Reuse Support Structure," *Proceedings of the IEEE International Conference on Communications*, IEEE Service Center, Piscataway, NJ (June 1991), pp. 1500–1504.
2. R. Prieto-Diaz, "Implementing Faceted Classification for Software Reuse," *Communications of the ACM* **34**, No. 5, 89–97 (May 1991).
3. S. T. Redwine and W. E. Riddle, "Software Reuse Processes," *IEEE Proceedings of the Fourth International Software Process Workshop* (1988), pp. 133–135.
4. J. F. Rockart et al., *A Primer on Critical Success Factors*, Center for Information Systems Research, Massachusetts Institute of Technology (1981).
5. K. Nagel, *Die 6 Erfolgsfaktoren des Unternehmens*, fourth edition, Verlag moderne industrie, Landsber/Lech, Germany (1991).
6. M. Wasmund, "Critical Success Factors of Reuse at IBM Böblingen, Germany," *Proceedings of the Fifth Annual Workshop on Software Reuse*, M. Griss and L. Latour, Editors, Department of Computer Science, University of Maine, Orono, ME 04469 (November 1992).
7. IBM uses a reuse maturity model to assess the progress of reuse technology practice within an organization. That model is derived from the maturity level defined in M. Pulk, B. Curtis, and M. Chrissis, *Capability Maturity Model for Software*, Technical Report CMU/SEI-91-TR-24, Software Engineering Institute, Pittsburgh, PA (1991).
8. S. D. Fraser, "Enhanced Reuse with Group Decision Support Systems," *Proceedings of the Second International Workshop on Software Reusability*, R. Prieto-Diaz and W. B. Frakes, Editors, IEEE Computer Society Press, Los Alamitos, CA (March 1993), pp. 168–175.
9. C. Pollard and C. Saunders, "The Case for Interpretive Structural Modeling as a Technique for Enhancing Electronic Meeting Support," *Proceedings of the Twenty-sixth Hawaii International Conference on System Sciences*, IEEE (January 1993).
10. J. W. Hooper and R. O. Chester, *Software Reuse, Guidelines and Methods*, Plenum Press, New York (1991), pp. 17ff.
11. B. Rothery, *ISO 9000*, Gower: Aldershot, second rev. edition (1993).
12. T. Davis, "Toward a Reuse Maturity Model," *Proceedings of the Fifth Annual Workshop on Software Reuse*, M. Griss and L. Latour, Editors, Department of Computer Science, University of Maine, Orono, ME 04469 (November 1992).
13. J. R. Tirso, "The IBM Reuse Program," *Proceedings of the Fourth Annual Workshop on Software Reuse*, L. Latour, Editor, Department of Computer Science, University of Maine, Orono, ME 04469 (November 1991).
14. W. Tracz, "Where Does Reuse Start?," *ACM Software Engineering Notes* **15**, No. 2, 42–46 (April 1990).
15. W. Tracz, "Software Reuse: Motivators and Inhibitors," *Proceedings of Computer Conference 1987* (COMPCON '87), IEEE Computer Society Press, Washington, D.C. (1987), pp. 358–363.
16. V. R. Basili and C. Gianlugi, "A Reference Architecture for the Component Factory," *ACM Transactions on Software Engineering and Methodology* **1**, No. 1, 53–80 (January 1992).
17. M. Lenz, H. A. Schmid, and P. F. Wolf, "Software Reuse Through Building Blocks," *Software* **4**, No. 4, 34–42 (July 1987).
18. K. Harris, "Using an Economic Model to Tune Reuse Strategies," *Proceedings of the Fifth Annual Workshop on Software Reuse*, M. Griss and L. Latour, Editors, Department of Computer Science, University of Maine, Orono, ME 04469 (November 1992).
19. W. C. Lim, "A Cost Justification Model for Software Reuse," *Proceedings of the Fifth Annual Workshop on Software Reuse*, M. Griss and L. Latour, Editors, Department of Computer Science, University of Maine, Orono, ME 04469 (November 1992).
20. K. C. Kang, "Features Analysis: An Approach to Domain Analysis," *Proceedings of the Reuse in Practice Workshop*, J. Baldo and C. Braun, Editors, Software Engineering Institute, Pittsburgh, PA (July 1989).
21. J. Margono and L. Lindsey, "Software Reuse in the Air Traffic Control Advanced Automation System," paper presented at *Joint Symposia & Workshops: Improving the Software Process and Competitive Position*, Alexandria, VA (April 29 to May 3, 1991).

Accepted for publication April 1, 1993.

Michael Wasmund *IBM Germany, Software Engineering Tools Department, Schönaicher Strasse 220, W-7030 Böblingen, Germany.* Mr. Wasmund is the site reuse champion at IBM's system software laboratory in Böblingen, focusing on implementation of a reuse methodology throughout product development. Previously, he was research staff member at IBM's European Networking Center, Heidelberg, Germany, where he published several papers about networking in heterogeneous environments. At the beginning of his IBM career, he was involved in the development of IEEE 802.3 adapters for System/370™ processors. He obtained his B.S.E.E. and M.S.E.E. degrees in Ulm and Bremen (Germany), respectively.

Reprint Order No. G321-5526.