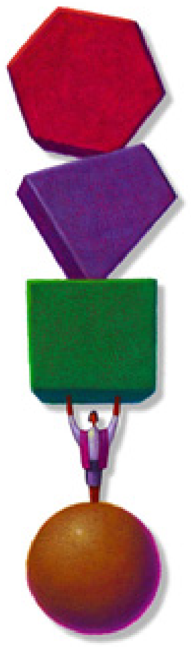
Principles of Process Improvement

by **Ian Spence**
Technical Lead
Process and Project Management Team
Rational Services Organization, UK

Nothing is more difficult than to introduce a new order. Because the innovator has for enemies all those who have done well under the old conditions and lukewarm defenders in those who may do well under the new.
-Niccolò Machiavelli, 1513 A.D.



Changing the software development process is hard enough when you are only addressing a single, small pilot project. It becomes exponentially more difficult when the target is a large project, program, department, or organization.

This article takes a look at what makes large-scale process changes so difficult, discusses the role of formal Process Improvement Programs in effecting organizational change, and presents a selection of experience-based principles that can be applied to improve a program's chances of success.

Process Change Is Organizational Change

Before we start to discuss Process Improvement Programs and their effects, we should take a step back and look at the basic concepts that shape any software development organization. The whole organization is based upon four fundamental concepts, as shown in Figure 1.

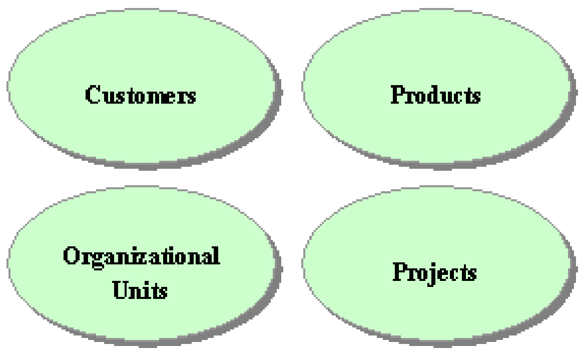


Figure 1: Concepts that Shape a Software Development Organization

There are a number of fundamental interrelationships among these four concepts:

- *Projects* create and change *Products*.
- *Projects* draw on resources from *Organizational Units*.
- *Organizational Units* organize resources and support *Customers*, *Projects*, and *Products*.
- *Customers* consume and commission *Products*.
- *Customers* provide the market and requirements for the *Products* produced.

Missing from this model is the *infrastructure* that binds all of these concepts together: the *Organizational Culture* (see Figure 2).

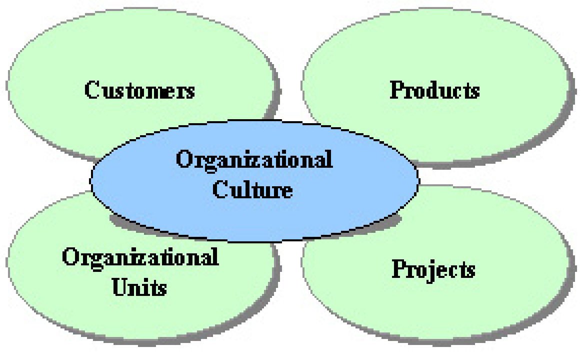


Figure 2: *Organizational Culture* Is the Infrastructure of a Software Development Organization

In its purest form, organizational culture can be thought of as the philosophy and practices that drive the form the organization takes, the products it produces, and the customers it attracts.

For organizations engaged in the business of software engineering, the organizational culture comprises a combination of the underlying business culture and business procedures, the larger company culture, operational procedures, and the *software development environment*. The latter encompasses all of the things a project needs to develop, deploy, and maintain a product, such as processes, tools, guidelines, templates, and technology.

It is also the area that most directly affects the software engineering business's performance, and the one most organizations choose to address when attempting to achieve performance improvements. The key message of this article is that *the software development environment, and by implication the software development process, cannot be changed in isolation from the organizational culture*. Any significant change to the development environment will, and should, have an impact upon the other areas of the organizational culture.

In fact, to produce lasting improvements in a software development organization's performance, we must:

- Embed the need for improvement directly into the heart of the organizational culture.
- Adopt a holistic approach that will continuously improve the support provided for all aspects of the organization.

Often, the most effective method for producing fundamental change within the software development environment is to change the underlying software development process. This means embarking on a program of organizational change.

Why Is Organizational Change So Hard?[1]

Organizations must change in order to keep pace with the changing environment around them. Living in denial and freezing the organization simply causes necessary changes to pile up until they cause a crisis. So why is organizational change so hard to achieve and crisis so prevalent?

Unfortunately, people resist change. In fact, whether because of inertia or fear of the unknown, people will often actively oppose change, or at least slow it down to what they feel is a manageable pace. In order to change an organization, you must understand these forces and channel them to enable, rather than oppose, change.

In his 1995 paper "Affecting Organizational Change,"[2] Roger Hebden expresses the forces of change as a simple formula:

pain + desire = change

The fundamental tenet of this formula is that change is ultimately driven by emotions. Pain and desire are the forces that drive us to make a change and to accept it. Pain is the catalyst that initiates a change, and desire is the force that pulls us toward a goal. A successful transition entails understanding and managing the perceived level of pain and the desire for a solution. This is what D.Connor calls *pain management* and *remedy selling*.[3]

Pain management means identifying and communicating what the fundamental issue is (our experience shows that the underlying problem is often in the organization's process or in the absence of one) and why change is necessary.

Remedy selling consists of two activities: solution selling and transition planning. It isn't enough to describe the ideal goal. To define a solution, you also need a path from where you are to where you are going, with some clearly defined intermediate milestones.

Change will not happen just because management says so. It is too easy to generalize about both problems and solutions. "Every project must follow the Rational Unified Process" is a sweeping generalization about how projects need to act; however, it does little to initiate change.

To successfully implement an organizational change, the adopting organization must:

- *Identify change agents* at various levels of the organization. This is the set of individuals who will take on the mission to make the change happen.
- *Formulate the real nature of the problem*. The change agents must understand the real nature of the pain and communicate it to the rest of the organization to raise awareness.
- *Plan the changes* in small, reasonable, and measurable steps describing both the goal, and the path to the goal. The plans must balance the need to achieve immediate, tactical improvements against the organization's longer-term strategic objectives.
- *Communicate the changes* in terms of tangible, quantifiable achievements and activities in a way that is understandable to all levels of the organization.

When any sort of large-scale organizational change fails, it can usually be ascribed to one of the following causes:

- Failure to incrementally implement the changes.
- Lack of management support.
- Lack of practitioner support.
- Lack of stakeholder support (i.e., customers, other departments, subcontractors, and suppliers).
- Lack of willingness or ability to deal with organizational change.

It is exactly these non-technical issues that derail many organizations' attempts to fundamentally change their software development environment.

Introducing the Process Improvement Program

Process changes affect individuals and organizations more deeply than changing technology or tools, since they strike at the heart of people's fundamental beliefs and values, changing the way they view their work and its value. In some cases they can even go as far as changing the organization's reward structure as well as the physical working environment, business culture, and politics. They also affect the way the organization works at the project level, the department level, and with other organizations. As Ivar Jacobson notes in *Software Reuse: Architecture, Process, and Organization for Business Success*,[4] this kind of process change amounts to "reengineering your software engineering business."

The management, planning, coordination, promotion, implementation, and measurement required to support this kind of change requires a formal Process Improvement Program.

This program needs to address the following areas:

- **People.** This includes their competencies, skills, motivation, and attitude; everyone needs to be adequately trained and motivated.
- **Projects.** The projects interacting with the Process Improvement Program must feel that they are receiving direct benefit from their involvement, so they should understand both the potential effectiveness and the costs of the proposed changes.
- **The process model.** This should define dependent structures, activities, and practices as well as artifacts to be produced.
- **Distribution mechanisms.** This includes how the process will be described, promoted, and distributed.
- **Supporting tools.** New tools will inevitably replace old ones, and this will require customization and integration.

It should also take into account the broader stakeholder community, including:

- **Managers** who are responsible for the performance of the software development organization. Any Process Improvement Program must have executive support, so these people must understand why the process is being changed, the potential return on investment, when and how progress is being made, and that expectations need to be carefully managed.
- **Customers**, both internal and external to the company. These people may need to be informed that the organizational process has changed, because it could affect how and when their input will be addressed and how products will be delivered.
- **Other parts of the organization** may also be affected. Sometimes changes in one sector of a business may lead to resistance and skepticism from other sectors that may not understand the reasons for the changes. Even if they don't have a direct influence on the program, ignoring other parts of the organization may cause political problems.

The Process Improvement Program is primarily an agent of organizational change. Its primary method for facilitating change is by improving the underlying software development process and its supporting tool set.

An Experience-Based Approach to Process Improvement

Experience also shows us that a number of basic principles can be applied to improve the Process Improvement Program's chances of success:

- Run it as a business program -- think "return on investment."
- Understand the existing environment.
- Build on the best -- don't reinvent the wheel.
- Make measurable, incremental improvements.

- Focus on projects and products, not organization.
- Use mentors, not enforcers, to facilitate rather than obstruct.
- Distribute process ownership.
- Keep people informed and involved.

The rest of this section looks at each of these principles in more detail.

Run It as a Business Program -- Think "Return On Investment"

Manage and plan the Process Improvement activities just as you would all the other activities in the business: Set up milestones, allocate resources, and manage and follow up as you would for any other program. Think "return on investment" when scoping activities; focus on those things that will pay back more than what was invested.

Some Process Improvement Programs devote too much time and too many resources to developing extensive guidelines, customized processes, and additional process-related material before beginning a project. There are four major problems with this:

- People do not read extensive descriptions.
- Doing everything right from the beginning is very difficult. It's better to try out something small, adjust it, and then expand the scope.
- People with process knowledge should spend most of their time mentoring, not writing extensive descriptions.
- These activities produce no measurable return on investment. Creating process definitions will not improve anything unless you apply the process.

No organization can afford to wait years for the perfect process to be fully documented before realizing actual improvements or a return on investment. The Process Improvement Program, whatever its ultimate aim, must include a substantial payback on the projects it affects.

Understand the Existing Environment

Before embarking on any major Process Improvement Program, the current development environment must be clearly understood, including:

- Current processes and practices
- Current tool set
- Current roles and team structures
- Products to be supported and produced
- Organizational process maturity
- Dynamics and politics of the current organizational structure

Without establishing this baseline understanding, you cannot make decisions about which process improvements should have highest priority and how to measure improvements.

Build on the Best -- Don't Reinvent the Wheel

The Process Improvement Program needs a firm foundation to build on. There are three sources available to build this from:

- Things that the organization does well. Harvest best practices from the most successful projects.
- The leading practices and products of the industry.
- The experiences of the people involved in the program.

Remember that the objective is not to transform the environment in one fell swoop, but rather to improve things bit by bit. To this end, the Process Improvement Program must focus on the organization's core business, prove all proposals in conjunction with real projects, and ensure that it is improving, not degrading, the performance of the business.

Make Measurable, Incremental Improvements

Implement environment, process, and tool changes incrementally to avoid overwhelming project teams with too many new challenges all at once. Introducing a new environment one piece at a time will increase the probability of success, and if you have assessed the development organization carefully, then you will know which parts of the process to introduce first. Typically it is best to start with the most problematic areas.

For example, an appropriate approach might be to:

- Focus the first iteration on improving the way requirements are captured and managed.
- Focus the second iteration on the analysis and design of applications and their components.
- In subsequent iterations, develop and deploy more and more parts of the environment.

Focus on Projects and Products, Not Organization

Too many process improvement initiatives focus on the organizational structure to support the desired process before the process itself has been proven or accepted. The thinking is, "If I put all my pieces in the right places, then they will automatically work in the right way and produce the right things," or "If we build it, they will come." In fact, this approach is naive at best and disastrous at worst.

The organizational structure should depend on the process and not vice versa. The process you select should depend on the types of projects your organization undertakes and the types of products you produce. You can establish an optimal organizational structure (an objective of the Process Improvement Program) only when a process is in place. *The required organizational structure should grow organically from the overall success of the Process Improvement Program*. In fact, to develop, establish, and deploy the products of the Process Improvement Program, a number of interim, transitional organizational structures are usually required.

The criteria for judging the effectiveness of an organizational structure are the same as those for the Process Improvement Program: the quality of the products produced and the efficiency of the projects that produce them. Focus on the products and projects within the organization before attempting dramatic restructuring of the organization itself.

Any process recommended by the Process Improvement Program must provide a firm foundation for the continued evolution of the organization's product set. The process must be scalable and capable of continuous improvement and evolution alongside the business that it supports. As the business grows and changes, the process will start to dictate and mold the underlying organizational structure.

Most successful Process Improvement Programs start as small, focused projects working within the existing structure. Remember: Consultants, centers of excellence, tiger teams, hit squads, and so on, are just tools to be deployed as part of the ongoing Process Improvement Program.

Use Mentors, Not Enforcers, to Facilitate Rather than Obstruct

Mentors act as drivers of change. Experience shows that using mentors is crucial if you want the implementation of a new process to succeed. Without them, there's a clear risk that people will fall back into their old habits.

Allocate sufficient resources, and plan mentoring activities carefully. The Process Improvement Program needs both the resources and the budget to enable the mentoring of the projects adopting and configuring the process. The cost of adopting the new process must not fall solely upon the adopting projects. As well as supporting the project's development staff, it is important that a process mentor works with, and supports, the project manager to ensure that the process changes progress in synergy with project work.

The mentor should focus on transferring both knowledge and responsibilities to project members and ultimately become dispensable and obsolete. Mentors must also remember their responsibilities to the Process Improvement Program, including:

- To harvest best practices from the projects they support.
- To make "just in time" process improvements and configurations.
- To promote the Process Improvement Program within the projects.

Some organizations adopt a hands-off, review-led approach to process improvement. Instead of actively engaging with the projects to promote and facilitate adoption of the new process, they rely on threats and reviews to ensure compliance. This "enforcer" approach is fundamentally flawed; the involvement nearly always comes too late for the projects adopting the new process and consequently obstructs, rather than facilitates, adoption as well as project progress. In many instances, without a mentor as process driver, the whole implementation fails.

Distribute Process Ownership

Distributing ownership of the process among project members gets them more involved, reduces their opposition to the changes, and allows their experiences to be leveraged to the benefit of the whole organization. The resulting process is better when the "real experts" -- people actually working on the project -- develop it themselves. Distributing process ownership reduces the likelihood that projects will become too heavily dependent on outside consultants and helps project members feel like part of the program rather than its victims.

As soon as possible, appoint people on the projects to be responsible for each in-scope, core area of process improvement. They should be people who know the area well, can mentor other project members, and will champion the Process Improvement Program within the project.

They should have primary responsibility for configuring and publishing individual parts of the process, and they should also help those who own the different parts of the process to define, configure, and document the process.

Keep People Informed and Involved

The greatest threat to any organizational change is people's natural resistance to change. Introducing new processes and tools means that people have to modify the way they work, and many grumble about it, or worse. This creates the risk of a negative spiral: Negative attitudes can lead to poor results, which then lead to even more negative attitudes, and so on.

Some actions you can take to prevent negative attitudes from taking over are:

- **Set realistic expectations.** Do not oversell the new process or the new tools.
- **Explain why the change needs to happen.** Makes sure people know what problems the organization has that need to be solved. What changes in technology require a new process and new tools? How will people benefit from using these new tools and process?
- **Inform everyone in the organization about what is happening.** This information doesn't have to be very detailed; the important thing is to make sure people know about the changes.
- **Remember stakeholders, such as customers or sponsors.** If you are changing from a waterfall to an iterative development approach, for example, the stakeholders must understand how an iterative development project is managed and how progress is measured. With an iterative development project, they shouldn't expect a completely frozen design at an early milestone, for example, and they should also

know that requirements may be captured differently.

- **Involve key people in the change work.** Give them responsibility for parts of the process.
- **Educate project people about the new process and tools.** After all, they will need to understand both how the new process works and how to use the new tools.

The Process Improvement Program Is an Agent of Change

The Process Improvement Program is a change mechanism, not a department. Although it will initially have responsibility for supporting and maturing the software development environment that it produces, the Program must resist the temptation to overcentralize and start empire building.

As we have seen, the major issues involved in successfully effecting organizational change are political and personal rather than method-, tool-, or technique-related.

The program's management should keep in mind that:

- **The support organization may already be in place.** Most organizations already have teams charged with supporting components of the software development environment. These people have knowledge and experience that will be essential to implementing the Process Improvement Program effectively, and they should be brought onboard as soon as possible.
- **Don't attempt to roll out changes before they have been established and proven.** Many programs set up a mass of central teams and attempt to mandate the way that people should work *before* the process itself has been established or proven. For example, many organizations spend a great deal of money setting up and building reuse repositories before they have even proven their capability to produce and maintain any formal reusable assets.
- **Don't get carried away with your own publicity.** Transforming an organization takes time. The ultimate result of the Process Improvement Program may be to transition all teams to component-based development, or to set up the structures of a reuse business. But these things will happen only if they will result in overall process improvements, and they must be approached step by step.
- **Any centralized solution is only a framework.** As the software development environment matures, it is very tempting to start believing that there is a "one-size-fits-all" solution that can be mandated for all existing and future projects. Different areas of the process vary in response to different forces: The project management process may vary with the size and formality of the project, the design and implementation process with the technology chosen. Any centralized process must provide a common environment for the successful execution of projects while preserving the autonomy that the

projects need to be able to progress effectively.

Individual projects will also need to tune the process within the constraints laid down by the organizational culture. The process must encourage this, and provide a mechanism for the individual process improvements to be fed back to the Process Improvement Program and made available to the rest of the organization.

- **Bootstrap the software development environment** *(i.e., "Eat your own dog food")*. Use the concepts, methods, tools, and techniques of the proposed solution to build the solution itself. For example:
  - If you are proposing the introduction of business modeling techniques, then prove these techniques by modeling the software engineering organization.
  - If you are proposing that projects manage their requirements using a specific process and tools, then use these to manage the software development environment's requirements.
  - If you are proposing a specific project management approach, then use this approach to manage the Process Improvement Program.

In fact, this advice applies to most Program areas: Perform all activities using the new process that you want the organization to follow.

The Process Improvement Lifecycle

If you are truly successful in your Process Improvement Program, then process improvement itself will become one of the essential, underlying tenets of your organizational culture. The Process Improvement Program may never end; it will continue iterating through the basic cycle shown in Figure 3.

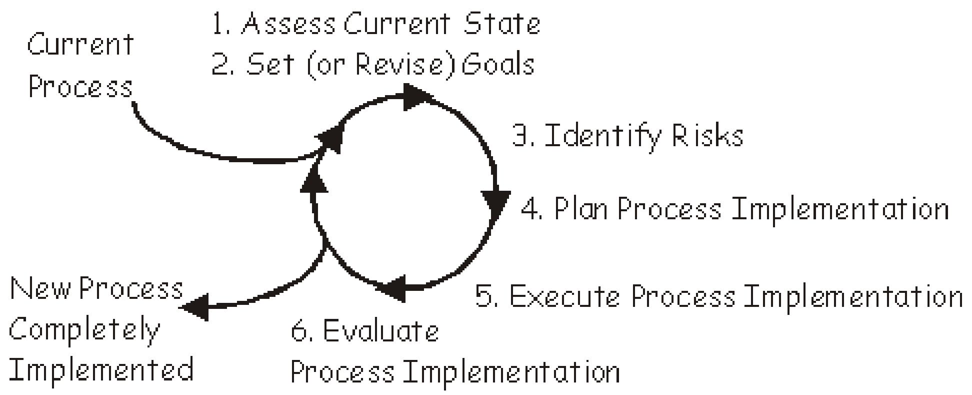

Current Process
1. Assess Current State
2. Set (or Revise) Goals
3. Identify Risks
4. Plan Process Implementation
5. Execute Process Implementation
6. Evaluate Process Implementation
New Process Completely Implemented

Figure 3: Process Improvement Program Phase Lifecycle

The Process Improvement Program itself consists of a number of phases, all of which follow the cycle shown in Figure 3. Each phase encompasses planning and implementation for a specific set of process improvements within, or extending, the underlying process framework. As the process establishes itself as a fundamental part of the organizational culture, the whole organization should move into a state of continuous process improvement. See Figure 4.

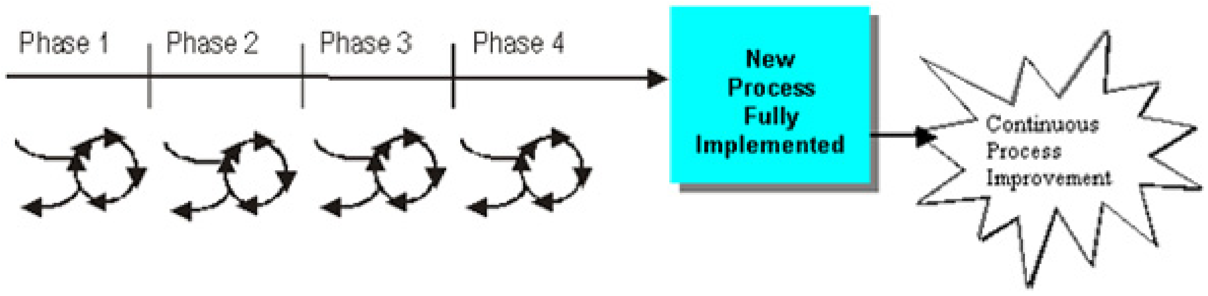


Figure 4: Continuous Process Improvement

The early phases will focus on establishing the software development environment. Once that has been fully implemented, the later phases of the program will be concerned with tuning, maintaining, and supporting it. The quest for process improvement should be ongoing: We should always strive to learn from our experiences and continually evolve and improve both the process and the organizational culture.

The size, number, style, and duration of the phases within the program depend on a detailed knowledge of the organization being addressed, the state of its current processes, and the scope of the Process Improvement Program itself. The RUP contains detailed guidance and illustrations of typical process implementation patterns and approaches.

Another essential source of guidance is the Carnegie Mellon Software Engineering Institute. In addition to publishing the Capability Maturity Model® for Software, which itself acts as a framework for undertaking process improvement, the Institute has developed the IDEAL model to guide organizations in planning and implementing an effective software Process Improvement Program.[5] Based upon the Capability Maturity Model, this model presents a more formal, and thoroughly documented, iterative lifecycle than the simple model shown in Figure 3.

Summary

To produce long-lasting improvements in software development organization performance, we must embed the need for improvement directly into the heart of the organizational culture and adopt a holistic approach that continuously improves support for all aspects of the organization. This means embarking on a program of organizational change.

The most effective method for producing fundamental change within a software development environment is to change the underlying development process through a Process Improvement Program. The Program should be set up and managed like any other business program, with a strong focus on return on investment and several phases of implementation.

Strategies for success include the following:

- Kick-start the Process Improvement Program by focusing on problem areas for which you can easily achieve positive results and people can see benefits early on.

- Communicate the problems with today's organization; if people understand those problems, they will accept the need for change.
- Do not try to do everything at once. Instead, divide the implementation into a number of increments and, in each one, implement a portion of the new process together with supporting tools. Focus on areas in which the change will have the most impact.
- Build on the best. Don't throw away all the things the organization does well. Integrate these into whatever standard process you choose.
- Implement the process and tools in iterations of actual software development projects to verify the process and the tools early on in a real environment. The process must improve and evolve in partnership with projects that will use it.
- Distribute ownership of the process among project teams. This will produce a better process, get people more involved, and reduce resistance to change in the projects.

The project team members assigned to the Process Improvement Program should:

- Mentor and facilitate projects adopting the process.
- Work closely with projects to capture and document appropriate process improvements.
- Promote the new process within the organization.

They should not attempt to draft a perfect one-size-fits-all process. The process will always require local configuration and provides three things:

- A framework for projects to quickly instantiate their process.
- A common basis and minimal acceptance criteria for all projects.
- A library of tools and techniques that projects can apply.

Implementing a new process, new tools, and possibly new technology makes a software project's schedule more volatile. Be sure to allocate enough time and resources to implement the process, train people, and so forth during all iterations. And remember: Keep all stakeholders well informed and involved as you make progress with your Process Improvement Program.

References

Books and Articles

D. Conner, *Managing at the Speed of Change*. Random House, 1992.

J. Jellison, *Overcoming Resistance: A Practical Guide to Producing Change in the Workplace*. Simon & Schuster, 1993.

Ivar Jacobson, Martin Griss, and Patrik Jonsson, *Software Reuse: Architecture, Process and Organization for Business Success*. Addison-Wesley, 1997.

Roger Hebden, "Affecting Organizational Change." Rational Software, 1995.

Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model for Software, Version 1.1." Software Engineering Institute, CMU/SEI-93-TR-24, DTIC Number ADA263403, February 1993.

Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, and Marilyn W. Bush, "Key Practices of the Capability Maturity Model, Version 1.1." Software Engineering Institute, CMU/SEI-93-TR-25, DTIC Number ADA263432, February 1993.

Robert McFeeley, *IDEAL: A User's Guide for Software Process Improvement*. Software Engineering Institute CMU/SEI-96-HB-001, February 1996.

Rational Unified Process

This article draws very heavily from the following sections of the Environment Discipline within the Rational Unified Process:

- Concepts: Effect of Implementing a Process
- Concepts: Mentoring
- Concepts: Managing Organizational Change
- Concepts: Environment Practices
- Concepts: Implementing a Process in an Organization

The Carnegie Mellon Software Engineering Institute

For more information on Process Improvement Programs and information on the Capability Maturity Model for Software and the IDEAL model, visit the Carnegie Mellon Software Engineering Institute Web site:

- http://www.sei.cmu.edu/ideal/ideal.html
- http://www.sei.cmu.edu/cmm/cmms/cmms.html
- http://www.sei.cmu.edu/sei-home.html

Notes

1 This section is extracted from the Rational Unified Process: Environment/Concepts/Managing Organizational Change, which in turn draws heavily upon Roger Hebden's paper, "Affecting Organizational Change." (See #2 below.)
2 Roger Hebden, "Affecting Organizational Change." Rational Software, 1995.
3 D. Conner. *Managing at the Speed of Change*. Random House, 1992.
4 Ivar Jacobson, Martin Griss and Patrik Jonsson, *Software Reuse: Architecture, Process and*

Organization for Business Success. Addison-Wesley, 1997.
5 See Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model for Software, Version 1.1." Software Engineering Institute, CMU/SEI-93-TR-24, DTIC Number ADA263403, February 1993 and Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, and Marilyn W. Bush, "Key Practices of the Capability Maturity Model, Version 1.1." Software Engineering Institute, CMU/SEI-93-TR-25, DTIC Number ADA263432, February 1993.

For more information on the products or services discussed in this article, please click here and follow the instructions provided. Thank you!