

Process Automation in Software Development

A White Paper



Abstract

Process plays a crucial role in the success of any software development initiative. Software development organizations can maximize the return on their people, tools, and technology investment through effectively selecting and managing their end-to-end processes. But the complex nature of software development makes manual management of the underlying processes impractical or prohibitively expensive. Automating these processes in a systematic and well-planned manner allows development organizations to achieve the benefits of process management in a practical and cost-effective manner. This white paper describes the role and importance of process automation in software development and outlines the key considerations in implementing effective process automation strategies.

Content

- Unique Nature of Software Development 2
- Process in Software Development 3
- Importance of Process 4
 - Process Makes Methodology Real 4
 - Quality Plans Demand Well-Planned Processes 4
 - Process Provides a Context for the Effective Use of Tools 4
 - Process is Key in Outsourcing and Offshore Development 4
- Implementing a Cohesive Process Plan 5
 - Process Selection 6
 - Process Modeling 6
 - Process Enactment 7
 - Process Monitoring 7
- Automating Software Development Processes 8
 - Process Automation Systems 8
- Alternatives to Process Automation 10
 - Extend the Use of Development Tools 10
 - Utilize a Business Process Management System 11
 - Use Project Management Practices and Tools 11
- Conclusions 13

Unique Nature of Software Development

Software has permeated almost every aspect of business in all industries. However, development of software applications remains inherently complex and relies heavily on the knowledge and skills of high-paid professionals. In contrast to most other business domains, the people-intensive nature of software development has limited the extent to which process automation practices can be used to increase productivity and lower costs.

Because software development is complex, with multitudes of highly interdependent activities, making mistakes is typically very expensive. Despite the increased emphasis on quality and skills certification programs, mistakes do occur and frequently - an inevitable consequence, given the level of human involvement. When mistakes occur, they have wide implications. Misinterpreting a single business requirement at the design phase, for example, or interpreting a seemingly subtle aspect of the design at the coding phase, can lead to a significant negative impact on the project cost or timelines.

Application developers have used four key strategies to cope with this dual challenge:

- Adopting new **methodologies**
- Implementing **quality programs** such as CMM, Six Sigma, ISO
- Deploying **developer tools**
- Resorting to **offshore and outsourcing models**

These strategies have made a notable impact on organizations' ability to cope with the complex and people-intensive nature of software development, but they are not nearly enough. What seems to be missing from the mix is a solution for the implementation and management of the common denominator in all four strategies, namely the underlying *processes*. After all, the full benefits of embracing new methodologies, quality programs, developer tools and promising new business models can be realized only if the organization is able to manage the processes that these initiatives rely on so heavily.

Companies that devise, implement and automate the right processes across their development organizations can leverage the above strategies fully, and in this way meet and exceed their quality, time-to-market and cost goals. But before examining the best process practices, it is worthwhile to arrive at a clear understanding of the definition and implications of the term process.

Process in Software Development

Broadly speaking, process defines the manner or approach in which software development takes place. It defines who performs what, in what order, to produce what. A well-defined process includes a clear definition of at least the following five items:

- Roles
- Skills
- Activities
- Work products
- Relationship between the activities that have complex dependencies

Software development teams have an existing process in place. The present development process may be *tacit* in that it has never been explicitly defined, documented and communicated. But it exists nevertheless; how else can the development team be producing its software application?

Alternatively, the present process may be *explicit*, in that it is clearly documented and understood by developers and management.

As outlined in subsequent sections of this document, a number of important benefits accrue to organizations that define and manage their development processes. However, those benefits can only be realized if process planning is approached in an explicit and conscious manner. Organizations' processes can be refined, managed and automated, but only if they are first defined, documented and communicated explicitly.

Process plays an important, albeit different, role in every form of software development. Development of the embedded software that monitors and controls the many electronic systems in a modern car requires sound processes as does the development of a packaged enterprise application. Notwithstanding its universal relevance and importance, process takes on different forms in different development environments.

Custom processes that are tailored to meet one organization's unique requirements still represent the majority of the market, despite the recent rise in common "proven methodologies" and "best practices". Irrespective of the type, when development processes are explicitly defined and actively managed, they lead to significant benefits.

Importance of Process

Improved process management practices help development organizations cope with complexity, inefficiency, human error and undesirable shortcuts - all real and prevalent issues in software development. More specifically, a well constructed process plan benefits application development in four ways.

- Benefit 1: Process Makes Methodology Real
- Benefit 2: Quality Plans Demand Well-planned Process
- Benefit 3: Process Provides a Context for the Effective Use of Tools
- Benefit 4: Process is Key in Outsourcing and Offshore Development

Process Makes Methodology Real. Development organizations need well-orchestrated processes if their adopted methodologies are to produce the intended results. The adopted methodology may entail proven and respected practices in the industry and be tailored perfectly to meet organizations' particular requirements; yet it will remain conceptual, unless and until the underlying processes are modeled to reflect the methodology's characteristics and these processes are enacted properly (see sidebar "Process vs. Methodology").

Quality Plans Demand Well-planned Processes. Process plays a crucial role in quality programs such as CMM, Six Sigma and ISO. Large development organizations, in particular, are increasingly subscribing to these programs to help them adopt a systemic approach to quality across their organization. Beyond the inherent benefits, the organization also gains external industry recognition from being certified at increasingly demanding "levels".

Process plays a mandatory role in all these programs. For example, implementing sound process implementation and management plans is an integral part of every CMM level. The initial levels require development processes to be explicitly documented and communicated. Higher CMM levels go as far as requiring proactive auditing of development processes.

Process Provides a Context for the Effective Use of Tools. A company can maximize their existing investment in specialized functional tools when the right processes are devised and implemented. Development tools for requirements management, software modeling and test automation, reduce or eliminate many of the mundane, repetitive and error-prone tasks performed by developers in each discipline.

The usage pattern of each tool and the manner in which the developer using the tool interacts with other developers in the team are governed by the process being used. Careful design of the end-to-end development process, and sub-processes that govern the activities within each discipline, can have a positive impact on the return on investment of existing functional tools.

Conversely, the development process would influence the selection of development tools. In either case, process provides the right framework for the effective use of tools.

Process is Key in Outsourcing and Offshore Development. A well-designed process management scheme could have a pivotal role in the success of outsourcing and offshore development initiatives.

The recent trend toward outsourcing enterprise application development to specialized software development outsourcing is driven by cost pressures and the need to focus on the core business. Software development outsourcing has become an accepted norm for small and large enterprises.

Process is key to the success of any application outsourcing contract, because outsourcing requires a close collaboration between two distinct businesses, each with its own set of approaches, priorities and skill sets. In many outsourcing engagements, only some of the application development work is outsourced (out-tasked) to the external vendor, necessitating an even closer interaction between various functions. Even in the case of full outsourcing, project

Process vs. Methodology

The relationship between process and methodology in software development is often confusing. In its most common use, methodology refers to a specific prescribed and disciplined approach to developing software. Developing software using a methodology involves planning and upfront consideration with a goal of increasing the efficiency and predictability of the project. It is the opposite approach to developing software spontaneously, sometimes even chaotically. Methodology-based development has attracted a considerable amount of attention over the recent years. Some of the well known categories of methodologies include agile or "lightweight" methodologies, iterative methodologies and waterfall methodologies.

Process, on the other hand, is the collection of all the detailed and explicit steps that the development organization needs to take in order to make the adopted methodology real. For example, without a clear definition of roles, activities, and work products—key aspects of defining the process—the selected methodology cannot be put into practical use. It remains a conceptual approach without any real and practical benefits to the organization. Therefore, process and methodology go hand-in-hand. Process is needed for the realization of methodology and methodology is needed to guide the design of the process.

success is dependent on how well the roles, activities and work products are coordinated among the customer and the vendor at a few key interface points. Deployment and management of defined processes that span both parties is essential to the success of the project.

As outlined previously, development processes need to be modeled to support the choice of development methodology. Enterprises that outsource their development should have a significant say in both the determination of methodology and the supporting processes. Often an organization's extensive knowledge of its application provides invaluable insights that can only have a positive impact on the process design. At the same time, an outsourcer's broad exposure to various customer projects is instrumental to match the right methodology and development processes to an organization's project. The ideal process model is likely different from what either party would have used if it were to develop the application in isolation.

In addition to process design, outsourcing organizations need to implement process management and automation schemes that, on the one hand, are suited to span across the two organizations, and on the other, allow the customer to maintain meaningful control over the project. As with process design, ideally, the selection and use of process management practices and automation systems would be done by the parties using a joint collaborative approach.

In a similar manner, organizations that use an offshore development model to extend their time-to-market and cost advantages, need to devise and execute process management strategies that are in

accordance with the distributed nature of the model. In most instances, some portion of the development work remains onshore, necessitating process models that can accommodate the resulting work breakdown structure and distance limitations. The addition of distance places an added emphasis on the organization's process management and automation approach.

Implementing a Cohesive Process Plan

If process plays such a pivotal role in handling complex and people-intensive software development, how do development organizations approach planning and implementation of this critical facet?



Figure 1

The four-step model depicted in Figure 1 presents one promising approach. In this stepwise approach, the organization begins by selecting one or more explicit processes that support its chosen methodology, which in turn is determined by the nature of the applications it is charged to develop. A visual and detailed model of the designated process is developed, disseminated and internalized by the development team through various programs of communications, awareness and training. The organization then enacts the modeled process by adjusting the roles, deliverables, development tools, and ideally, the compensation plan of the team in accordance with the requisites of the modeled process. Finally, the fully enacted process is monitored both to enforce the process and to determine legitimate areas where adjustments to the process model are needed in light of real experience.

Process Selection. Selection of the right processes either takes the form of making the existing processes more explicit or choosing new processes that reflect the future state the organization wants to emulate.

Many development organizations are satisfied with their existing development processes. These processes have typically evolved over a long period of time to suit the organization's specific strengths and limitations and the particular applications they develop. The challenge faced by these organizations rests more in enforcement and monitoring existing processes than selecting a new process. In such cases, process selection takes the form of simply specifying the characteristics of the existing process in clear explicit terms before moving on to process modeling. Further consideration of which methodology needs to be adopted is unwarranted, since the organization is a step beyond that.

Other development organizations have decided to select a new development process. The organization may have chosen a development methodology and needs to select processes that support this methodology. Or they may need to make certain improvements to existing processes.

In these cases, the task of process selection begins by determining which development methodology meets the organization's requirements. Detailed description of factors used in choosing development methodology are beyond the scope of this paper. The key consideration is to ensure that the chosen methodology suits the characteristics of the applications being developed. Organizations developing many applications with differing characteristics may have to adopt more than one methodology.

Once the methodology decision is made, the organization can select the development processes that must support each methodology. At this stage, the organization simply lists the required processes and denotes the characteristics of each process (similar if not identical to the characteristics of the supported methodologies). Ideally, the specified processes span the full development cycle.

Process Modeling. Once end-to-end processes that support the chosen development methodology are selected, the organization must comprehensively model each process in detail, ideally using an industry standard.

Until recently, the job of modeling a software development process was complicated by the lack of open industry standards. Where a solution existed, it was in the form of a proprietary vendor solution, only available through the expensive engagement of a professional services organization. The outcome of a successful process modeling engagement was indeed a model of customer's development process, but captured in a format, notation and terminology that made little sense outside the organization. The consequence of being locked into one vendor's proprietary approach spilled into the subsequent process enactment and monitoring phases.

The recent introduction of the SPEM (Software Process Engineering Metamodel) specification by OMG (Object Management Group) is a response to the pent-up demand in the industry for an open standard for modeling software development processes. SPEM was developed specifically to accommodate the complex and unique nature of software development processes. The specification uses many of the UML concepts familiar to software development professionals.

The right approach for this step in organization's process plan, therefore, is to model or describe the selected processes in SPEM. The resulting process models then need to be disseminated to all team members and each team member needs to be trained and provided with the necessary guides and support material to understand and internalize the objectives and workings of the intended process.

Process Enactment. Enactment or implementation of the development processes — by this stage explicitly identified and clearly modeled in SPEM — can take on many forms depending on the complexity of the processes and the size of the organization. These include developer education, new incentive plans and a sound project management approach. Process enactment is best defined in the context of actual projects.

Effective process enactment depends on well-planned communication and education, like process modeling. The difference is that these plans focus on the “how” as opposed to the “what” of the process. Practitioners and management are trained on how they can support various aspects of the process in the context of their function. Process best practices and lessons learned are categorized and prioritized by roles most impacted. The education programs may also train practitioners on the proper use of development tools to shore up the process being enacted. Finally, practitioners, who ultimately know best what factors affect their function, are encouraged to actively recommend ways to increase efficiency and effectiveness of the processes being enacted. Management are guided to establish the necessary means for these recommendations to be considered and acted upon.

Success of process enactment is also tied to organization’s willingness and ability to adjust its compensation and incentive program to reward developer practices that buttress its development processes and discourage practices that compromise them. The incentive measures often have a component that is tied to achieving certain training thresholds and other components that relate to completing activities and deliverables described in the process.

One important success criterion for enacting development processes is thorough project management. There are plenty of proven project management methodologies and best practices that have tremendous value to software development. In the early stages of enacting new processes, software development project managers should be encouraged to directly and explicitly focus on issues related to implementation of the process itself, not just the projects that use the process. For complex processes and large organizations this may entail assignment of dedicated process project managers.

The schemes described so far, go a long way to help the enactment of new processes in the organizations, but they are all manual, time involved and relatively expensive to implement. As described later in this paper, automation of software development processes offers a solution. Any software development organization with approximately 50 or more developers should consider automating its software development processes to reduce its overhead, cost and error margin.

Process Monitoring. Development processes that are enacted then need to be monitored closely. Monitoring or ongoing management of development processes allows the organization to determine if and to what degree the prescribed processes are being adhered to (being enforced). Process monitoring also helps to identify areas of improvement while the process is in actual use.

Software development organizations make process investments because well-executed processes help them develop better software, quicker and more cost-effectively. Therefore, monitoring quality, timelines and cost of the projects based on a given process are critical to process monitoring.

An organization’s process monitoring approach also needs to measure planned and unplanned deviations from the process model and risks and issues associated with each deviation. This provides the organization with valuable information on the effectiveness of its enforcement approach, and it will point to how existing processes can be enhanced.

As with process enactment, monitoring of complex processes in large organizations can be achieved efficiently only if these processes are automated. This is especially the case when development processes are to be monitored in real-time.

Automating Software Development Processes

The four step construct outlined above will lead to development and management of software development processes that support organization's chosen methodology. As stated, well orchestrated processes are an essential requirement in quality programs such as CMM, leveraging companies' existing investment in developer tools and facilitating outsourcing and offshore development models. Because of the manual nature of the task, despite the apparent benefits, managing organizations' processes is difficult at best and impractical at worst. The larger the organization and the more complex its processes, the more cumbersome it becomes to deploy and manage those processes.

In this regard, process automation offers tremendous promise. Organizations that properly automate the modeling, enactment and monitoring aspects of their processes stand to benefit greatly. Process automation, through the use of a capable process automation system, leads to:

- an easy to understand model of development processes that is accessible online;
- motivation and empowerment of practitioners to better support the enactment and enforcement of development processes; and
- real-time monitoring of key process metrics as well as relevant metrics of the projects that use those processes.

Process automation is a relatively new field in software development. In functional domains such as finance, sales force management, client relationship management, human resources and inventory control, process automation has taken root and resulted in significant tangible benefits. The recent introduction of SPEM by OMG has opened the path for doing the same in the software development space. Process automation systems that allow development organizations model their selected processes in SPEM and then enact and manage those processes across their development team, are key to actualizing organization's process plans.

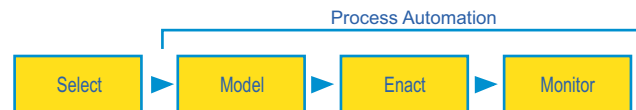


Figure 2

Process Automation Systems. As outlined above, process automation facilitates modeling, enactment and monitoring of development processes across the organization. As with any enterprise system, selection, deployment and proper utilization of a process automation system is a significant undertaking. Development organizations should look for the following key capabilities when considering a process automation system.

Enterprise Foundation: Development tools are typically single user or workgroup applications. In contrast, process automation systems are central applications that are used in real-time by almost all practitioners and development managers across the entire team. Strong enterprise class features, such as a multi-tier organizational hierarchy and sophisticated access control, are features that determine the system's scalability and use in large development settings. Furthermore, the system needs to have open interfaces for easy integration with back office systems such as resource management systems, ERP financial systems and ERP HR systems, as well as internal systems such as time tracking and skills management applications. Automation systems that

support the web services model and the associated protocols tend to simplify and reduce the integration challenge.

Pure Web Architecture: As with any modern enterprise application, a 100% web architecture eliminates all dependencies on costly client software rollout and management. Due to their visual and interactive nature, process modeling functions of the system are particularly hard to implement on a web interface. Automation systems that allow modeling, enactment and monitoring of complex processes through a simple web browser, without requiring the installation of a client application, have a direct positive impact on an organization's ROI.

SPEM Compliant: Process automation systems that provide 100% support for OMG's SPEM specification will prevent vendor lock-in by allowing the customer to use the modeled processes in other contexts. SPEM support at the enactment stage is particularly valuable and automation systems that use SPEM as their enactment framework hold particular promise.

Visual Modeling: Creation of process models is inherently demanding. Process automation systems that allow process engineers do so in a visual manner without resorting to any form of scripting tend to facilitate this otherwise difficult task. Feature rich systems allow process engineers to create new process models from scratch, modify existing process models, or fuse blocks of existing models to create a new model with ease. Visual modeling capability should not dilute the system's ability to create and modify complex relationships between various process elements.

Skills Management: Software development processes are inherently people-intensive, requiring highly specialized skills. Process automation systems need to manage roles, jobs, tasks and skills as an integral part of process automation. The development organization should be able to interface the system with the organization's existing skills management application, if one exists.

Concurrent Enactment Engine: Process automation systems' enactment engine should be capable of enacting multiple processes concurrently, each possibly using a different process model. The visible external aspect of the enactment engine manifests itself in the form of the practitioner interface. Developer acceptance is one of the main success criteria for any automation system. A practitioner interface that is configurable and burdens the practitioner with the least amount of extra work, beyond what the practitioner has to do in the normal course of his or her work, increases developer acceptance rate.

Seamless Interface to Point-Tools: Another aspect of enactment is related to interfacing process with development tools. Use of development tools in a particular discipline to generate one or more work products is an integral part of the development process. All tools have critical information repositories and this information must be shared across the development team. Here again, process flow defines the flow of this information. Automation systems that can interface with a broad range of functional tools in a seamless manner make the development processes more relevant to practitioners. The net result is improved developer productivity and enhanced developer acceptance.

Model Management: Process automation systems should be able to store modeled processes in a hierarchical process library. Ideally, the process model itself would be stored separately from process information such as process guidelines and best practices. Modification of the process model requires the skills of a trained process engineer, whereas a much broader base of users tend to contribute to process information. The system should include the capability to import and export any process between the process library and third party libraries that support the SPEM specification.

Real-Time Monitoring: Real-time monitoring of key process and project metrics by management is a key requirement for effective process management. Process automation systems that allow the organization, and even the individual manager, to configure the pertinent metrics, thresholds and

events tend to offer maximum value to the management team. Automation systems that offer a complete and sophisticated audit trail capability are indispensable in quality programs such as CMM.

Outsourcing and Offshore Development Features: Process automation systems that are designed from ground up to accommodate large, extended and distributed development teams can extend real benefit to programs that use an outsourcing or offshore development model.

Alternatives to Process Automation

So far, this paper has outlined why process should play a central role in any software development undertaking and described the importance of process automation and process automation systems in this regard. But is the deployment of a process automation system the only way to reap the benefits of process automation? Or, asked in a different form, can software development organizations use one or a combination of the existing tools at their disposal to achieve similar benefits?

Development organizations can perhaps resort to three different alternatives to assimilate the benefits of a full-blown process automation system:

Alternative 1: Extend the use of development tools

Alternative 2: Utilize a business process management (BPM) system

Alternative 3: Use project management practices and tools

As described below, although to varying degrees the above alternatives deal with one or more aspect of the process automation problem, none offer the full benefits of deploying an enterprise-level process automation system.

Extend the Use of Development Tools. A number of development tool vendors have built a limited amount of process management capabilities into their products in order to cope with process and workflow requirements (sub-processes) within the limited and specific software development discipline that their tool targets. Software Configuration Management (SCM) tool vendors, in particular, need to deal with a considerable amount of configuration and change management sub-processes. Similarly, source management tools need to cope with change management sub-processes.

Given the rising importance of process automation in software development, many of the point-tool vendors have started to position their products as process automation systems. However, despite the extensive use of process automation vocabulary in the products' marketing literature, these tools fall short of the minimum capabilities of a commercial process automation system in a number of areas.

As described elsewhere in this paper, software development processes are complex in nature. Automation of end-to-end processes often entails a far more difficult modeling and enactment challenge than trying to do so for a sub-process within a specific development discipline. Also, process automation systems are enterprise level applications, intended to be used by the full spectrum of practitioners and management. This is in contrast to point-tools that are typically designed as single-user or workgroup applications.

Development organizations should also watch for process modeling tools that may be mistaken for a full-fledged process automation system. Process modeling is clearly an important aspect of automation, but a small one. If the value offered by the tools ends at the point of merely publishing a visual model and a set of guidelines for the intended processes, the organization is still left with the challenge of enacting and monitoring these processes; a substantial portion of the process automation value equation.

Utilize a Business Process Management System. Business Process Management (BPM) is a new and emerging field in the IT industry. BPM systems allow common business processes to be modeled in a manner that would drive the implementation and deployment of the underlying IT applications and infrastructure. It is only natural to ask whether software development processes can be automated using a generic BPM system.

The limitation posed by applying a BPM system to software development also stems from the complex nature of software development processes. None of the present crop of BPM solutions tackles, or even targets, modeling and enactment of software development processes. A BPM system can be used perhaps to manage the simplest of software development processes, but none of the generic systems that belong to this category are able to accommodate complex software development processes.

Use Project Management Practices and Tools. Most large software development organizations use project management methodologies and project management tools to ensure their projects remain on time and on budget. Many have resorted to project management methodologies and tools to enforce and monitor their development processes. A number of project management tools allow project managers to create reusable project “templates” that may seem interchangeable for process models.

Some allow project participants to update the status of the tasks assigned to them; seemingly a form of process enactment. This raises the question of whether project management tools can be substituted for process automation systems, offering similar benefits.

As with the previous two alternatives, project management practices and tools can be stretched to deal at least with some aspects of processes automation, but with only limited benefits. There are four reasons why that is the case, none of which take away the importance and complementary role of project management.

Project manager centric vs. development team centric. The intended users and usage pattern of project management tools and process automation systems are fundamentally different. Project management tools are based on a paradigm where the project is centrally planned and managed by the project manager. The tool usage is based on a central paradigm and the main beneficiary of the tool is a central project manager. It is true that the new enterprise versions of some of the project management tools offer a capability whereby project participants can update their individual progress. But participants do so merely to simplify the job of the project manager.

In contrast, process automation systems are based on a process-centric paradigm where the project manager, just like any practitioner, is empowered by the system to enact certain aspects of the modeled process. Project management becomes another software development discipline.

Project template cannot contain the process model. A key benefit offered by a process automation system is its ability to develop the model of a process once and enact multiple projects using that model. Project templates offered by many of the project management tools may appear to be offering effectively the same capability. After all, a project template is created once and used multiple times in different projects. However, this apparent similarity is illusive.

The key difference is related to the nature of process model. Process model is fundamentally different from a project template in that it contains a complete portrayal of the process in the way that it is to be enacted. A project template cannot adequately capture or model the process using the fewer and more rudimentary constructs such as tasks and dependencies.

Process enactment and monitoring remain manual. The main benefit of process automation is elimination of manual intervention, especially in regards to enactment and monitoring aspects of the process. Manual steps are costly and error-prone.

In the absence of a process automation system, the only way that the project manager can help in the enactment and monitoring of the development processes is to contact the project participants regularly to determine if and how the process is followed (monitoring) and take steps to ensure the process is followed (enactment). The project manager would then have to manually update the project management tool with the latest information obtained through this manual method. In this way, the project management tool—which in the first place was designed as a planning tool—can provide the latest progress on each project, but only at the cost of much manual work.

The project management tools that allow project participants to update their progress through a web based interface seem to offer one way of reducing the manual steps. However, from a practitioner point of view, setting aside the time to update the assigned project tasks on a regular basis is usually considered a low value activity and a distraction to practitioner's core focus. In many cases, the practitioners simply stop updating the project management system, forcing the project manager to resort back to manual steps of calling progress meetings or contacting each practitioner manually. Even in cases where practitioners do take the time to update the tool, the information is often out of date barring the organization from the benefits of real-time process monitoring.

Leading process automation systems, on the other hand, include seamless interfaces with the developer tools familiar to and used by practitioners on a day-to-day basis. Far from taking time on the side to update a central and seemingly arbitrary project management tool (from the practitioner's perspective), practitioners end up playing a crucial role in the enactment and monitoring of the development processes by merely using the tools they have always used in their own discipline. Enforcement and real-time management of processes become a natural extension of what practitioners do in the normal course of carrying out their job.

Not intended for software development. Most project management tools are designed as generic tools and none aim to deal specifically with software development processes. A key theme stressed throughout this paper is that software development processes are unique due to their complex nature. Modeling software development processes—simplest of the three constituent elements of automation—using a generic business process modeling tool is formidable enough, let alone trying to do so using a generic project management tool.

Conclusions

The complex and people-intensive nature of software development demands that the right approach is taken to manage the underlying processes. Software development processes need to be defined in the context of the development methodology chosen by the organization. The choice of methodology, in turn, should be governed by the nature of the applications being developed.

Once defined, each process needs to be modeled, enacted and monitored. In development organizations that have 50 or more developers, the inherent and promised benefits of effective process management practices can be realized fully only if the organization automates its processes. Through automation, process modeling, enactment and monitoring become practical and cost-effective. Extending the use of existing project management and development tools into process automation will not produce the intended benefits.

Automation of software development processes is best accomplished through the effective use of a process automation system that is specifically built to fit the unique nature of software development processes. A leading process automation system must support open industry standards, be methodology and process agnostic, interface with existing development tools, and be able to enact multiple projects using different processes, concurrently.

Find Out More

To obtain additional information about SPEM, go to www.omg.org/technology/documents/formal/spem.htm or visit www.osellus.com.

About Osellus

Osellus is a leading provider of process improvement and automation solutions for the software development industry. Osellus helps software development organizations select, enhance, model and automate the underlying processes that support their application development efforts. By adopting a process-centric development model, Osellus customers are better positioned to meet their time-to-market, quality and cost objectives.



www.osellus.com
info@osellus.com

555 Richmond St. West
Suite 406
Toronto, Ontario
M5V 3B1 CANADA
Tel: (416) 603-6667
Fax: (416) 203-6000

10th Floor, 99/25 Moo 4
Chaengwattana Road
Klong Gleua, Pakkred,
Nonthaburi 11120 THAILAND
Tel: (662) 964-9810
Fax: (662) 962-2513