Providing Integrated Development Processes for Distributed Development Environments

Marco Kuhrmann, Georg Kalus
Technische Universität München,
Institut für Informatik – Software & Systems Engineering
Boltzmannstr. 3, 85748 Garching, Germany
{kuhrmann|kalus}@in.tum.de

ABSTRACT
As the world becomes smaller, distributed development becomes more and more common. In a distributed setting with many project participants being located at different places, methods and tools for coordination and collaboration are an imperative. Structured development processes of today usually address the issue of coordination. Collaboration on the other hand is one of the topics in modern development environments. In this paper we present first experiences with generative integration of development processes into development or collaboration environments. We present the concepts, techniques and sample implementations based on the German V-Modell®XT, which is the standard IT-development process for the German government services. Furthermore, we give an analysis of the current approach and discuss ongoing research.

Author Keywords
Development process, distributed development environment, model-based process-integration.

ACM Classification Keywords
D.2.2, D2.6, D.2.9, K.6.1.

INTRODUCTION

Distributed development seems to have become the standard rather than the exception. Reasons for this development are a lack of specialists at certain locations, differences in cost of personnel and resources, mergers and acquisitions, organizations serving local markets and many more [6]. A software development project doesn't necessarily have to be spread all across the globe to show characteristics of a distributed project. Studies show evidence that from a distance as little as 50 meters onwards, further distance between project participants doesn't matter anymore [2]. This would imply that a project with the team being spread over two buildings can and has to be classified as being distributed. Geographical separation of project participants has changed the face of software development projects and the way they have to be managed.

There are challenges particularly relevant to such projects. Among those are [6]:

- Communication/collaboration
- Coordination
- Synchronization (also across several time-zones)
- Knowledge management
- Differences in culture and language
- Allocation of work packages to teams at different locations
- Integration of work products to a combined result

Most of the difficulties have to do with the fact that without collocation, much of the informal communication and knowledge exchange usually taking place on the hallway or at the coffee machine is not possible. In a distributed setting, this has to be compensated for by appropriate methods and tools.

Software development and management thereof are difficult tasks in any case. The larger a software development project is, the more important is a development process that structures the project for it to be manageable. We claim that in a distributed setting explicit structure is crucial. Clearly assigned roles for example are vital for communication to work, because team members may not know all other team members and their responsibilities at a remote site. Also, to coordinate different teams at different locations, it has to be carefully defined for whom to deliver what and when.

Distributed development has gained popularity not at least because of the emergence of enabling technologies such as broadband internet that make it easier for teams to communicate and collaborate over large geographical distances. Many tool vendors have begun to capitalize on electronic communication technologies to create tools that form a centralized virtual workspace for distributed development.

While processes and tools are no silver bullet for successful distributed projects and certainly cannot (and are not meant to) replace face-to-face communication, evidence suggests that a distributed project not following a well-defined process and not using tools to facilitate communication and coordination is much more likely to not be successful [6]. In this paper we present our efforts at combining both worlds.

Structure

This paper is organized as follows: First we give an overview of related work. We then cover the area of process integration. We discuss meta-model-based processes and introduce the V-Modell XT as our working example. Next we discuss how and to what extent process models can be supported by a collaboration tool. We go on by presenting our integration approach. Based on that, we give an overview over our reference-implementations. We discuss their capabilities, our experiences and draw first conclusions. Finally we present open research questions and give an impression of our current work.

Related Work

Implementation of and tool-support for development processes has been discussed for a couple of years now [1, 3, 16]. Most of the approaches are based on graphs and graph grammars to provide a mechanism for integrating formal processes and tools. Westfechtel for example [4, 5, 10] showed some concepts especially designed for development processes. Some tool vendors such as IBM/Rational or Microsoft provide some tool-integrations that are in practical use. There are also some small and medium enterprises that provide specialized solutions for particular audiences (e.g. microTOOL or Polarion for the German market). The work cited above is more fundamental. Basic structures of processes are analyzed and combined with tool data models. Another research area of interest for our work is the investigation of processes at project-runtime. Project cockpits [20, 26] are used to improve controllability of a project by providing sophisticated measurement and analysis capabilities. Project cockpits collect heterogeneous data from a project and present them in a way that enables the management to easily grasp the project's overall state and trends. Furthermore, if something goes wrong in the project, a project cockpit can provide assistance for example in the form of de-escalation strategies, knowledge bases etc. Most currently existing approaches are reactive (passive) cockpits; meaning that they only analyze and give advice. If a cockpit had knowledge of the underlying process, one could imagine it to be proactive to a certain extent. Such a cockpit would not only depend on data provided by reporting engines, but would be controlled by a process. Trends could be identified and corresponding measures could be initiated. The third area of related work deals with the subject of process improvement and process integration. Here the provision of concrete, efficient methods for development processes is of interest [15, 27, 30]. This is especially important as standard processes like RUP [17] are gaining popularity.

In our work we put the best of those parts together: We use formal models of processes and tools to provide a flexible integration concept for process users. To optimally support them we provide project cockpit-like capabilities where possible and sensible. The tools we are creating are extensible, which enables users to consistently integrate micro-processes.

PROCESS-INTEGRATION CONCEPTS

Products such as IBM Jazz [7] and Microsoft Team Foundation Server (TFS) [8] are integrated tool-environments that aim at mitigating the aforementioned issues of distributed software development by helping teams to collaborate and synchronize and by offering a centralized virtual workspace. Yet we observe a gap between technically mature development environments and development process support. In the present tool landscape, development and management tend to be regarded as separate and rather independent disciplines. Generic collaboration platforms such as Microsoft SharePoint [11] do not imply a process at all while other tools, like IBM Jazz, impose a pre-defined and fairly static process on the organization using that tool.

As we find the necessary technologies to be readily available, we created a flexible and extensible set of generators integrating process models and collaboration tools. Before explaining our approach in detail, we outline the basic foundations for process-tool integration, specifically:

- Technical basics, especially meta-model-based processes, APIs of tools and structures of processes to be integrated in a tool.
- A sample process-model, which is meta-model-based and used in practice.
- Capabilities and limitations of process-tool integration.
- The addressed tools.

Meta-model-based Processes

We constrain ourselves to meta-model-based process-models. An informal or otherwise not machine-readable process would have to be brought into some kind of formal description before it could be mapped onto a tool with our approach. We therefore only address processes, which are already based on a meta-model, because such processes provide an API that can be used to extract elements and structure for conversion. Process-models based on a *formal* meta-model are usually known as *formal process-models* (sometimes they are also called heavy-weight processes as they extensively define structures, contents and dependencies).

Formal process-models are required as they not only declare process contents but also define them using a formal and machine-readable syntax, e.g. in the form of a XML-Schema. We require such a formal process description for the generators to be able to construct a mapping between the process elements and the data-model of the tool under consideration. The following elements of a process model are of particular interest for such a mapping:

- Product (or artifact) sub-model
- Activity (or task) sub-model
- Role sub-model
- Process sub-model

These sub-models together define the core of a typical process-model, which is usually a composition of artifacts, tasks and responsible roles. A meta-model-based development process can be seen as a blueprint for a project. Living macro- and micro-processes are instances of such a process meta-model. The relation between a formal process model and a project following that model is similar to the relation between classes and objects in the object-oriented programming paradigm.

Sample: V-Modell XT

In this paper we use the German V-Modell XT [12] as an example of a formal process-model. All reference implementations we have done so far are based on this process-model [20].

The V-Modell XT is provided by the German Ministry of the Interior as standard process for IT-development projects in the government services. It is a generic process that targets a wide audience and as such waives concrete methodologies (so called micro-processes) for specific tasks. The V-Modell XT requires customization in this area. It is an open model with open contents provided under the Apache License [13] including lots of (technical) documentation. Because the V-Modell XT is based on a XML-Schema-based meta-model (see online sources), tool vendors and content providers can easily process the model. Extensions, customizations and supporting tools can be provided in various areas.

In the following paragraphs we outline some basic concepts of the V-Modell XT and identify process elements that are suitable for tool support. The V-Modell XT is a so-called *product*- or *artifact-centered* process model, which means that products/project results are focused. Activities play a secondary role in the V-Modell XT. An activity is just there to finish a product. This is a major difference to processes based on SPEM [14], such as RUP [17] and EPF [18] as they focus on activities and tasks (*task-driven* process-models). Activities in the V-Modell XT are modeled relatively coarse-grained. An activity only specifies *that* a product has to be worked on but not *how*. This is a disadvantage if tool-integration is the matter of interest.

The concepts of the V-Modell XT that are relevant are:

- *Products* that are the results/outcomes of a particular process-step or a project.
- *Roles* that are responsible for a certain product or take part in its creation.
- *Activities* that act as work package to create a product. Each product in the V-Modell XT is created by exactly one activity, except for so-called *external* products, which are produced out of a project's context.
- *Project operation strategies* that describe the whole project's process structure. They define milestones (decision gates) and permitted paths to reach them. Each decision gate has a number of products assigned to it that have to go through quality assurance before the gate can be passed.

The V-Modell XT contains a set of predefined and customizable document templates for work products. Another feature of the V-Modell XT is its integrated *tailoring*-concept. The model defines rules and constraints that aid project managers during project set-up. Given a set of project characteristics, these rules regulate which parts of the process model have to be considered and what can be left out. With the help of a tool that adheres to those rules, the project manager can tailor the V-Modell XT to the needs of his project. The resulting tailored process is more specific and less voluminous (in some projects, the documentation alone is reduced to half [19] the size of the whole process documentation).

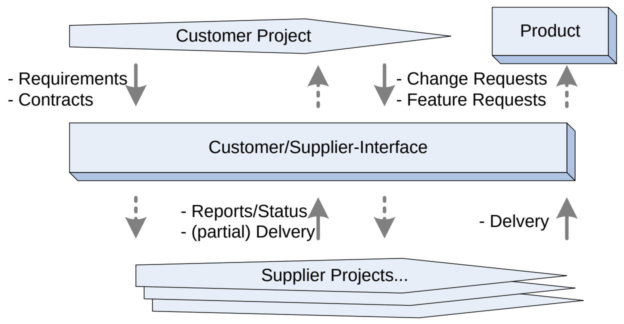


Fig. 1 Customer/Supplier interface of the V-Modell XT

The V-Modell XT is especially interesting for distributed development as it includes a fundamental role model for contracting and sub-contracting. V-Modell XT projects are *always* distributed projects because customer and supplier are integral parts of the model. The model defines an interface (Fig. 1) between all parties that regulates coordination between them. The interface consists of a set of artifacts (e.g. status reports) and ordered milestones that define who has to provide which deliverable and when. This concept is tightly integrated in the process- and organization-model. A tool that integrates the V-Modell XT would have that interface built-in by default and a project making use of it would fulfill it by definition. This has the great advantage that a remote project using the same process model can be seamlessly attached to. One could imagine a distributed scenario with a whole set of independent projects coordinating through the defined set of interface products.

For our approach we only need a subset of the V-Modell XT contents and structure. However, the process model alone is not enough. As mentioned above, the V-Modell XT elements are intentionally defined on a relatively abstract level. As we want to map the model onto a tool, we introduce some additional agreements. So, on the one hand we use the structural concepts and trim them according to our requirements. But on the other hand we also provide methodical additions to improve straightforward usability. These points are discussed in the next section.

Capabilities of Process-Automation

Looking at the structure of the process model we can identify several elements that are natural candidates for automation and tool-support for process enactment. As mentioned above, a process model typically consists of four main sub-models.

- Products or outcomes usually come in two shapes: Most of the work products are either plain documents or they are some kind of development or tool artifact, e.g. code, a design-model or a test case.
- Activities, at least as understood by the V-Modell XT, map straight onto tasks in a task list (usually represented as work items).
- Roles are an abstraction of people. In a tool or whole tool-infrastructure, this corresponds to user groups or the tool has some kind of role model anyway.
- Processes define which tasks have to be done in what order. In the world of tools we can identify at least two points of view: Firstly the *management*-viewpoint including schedule and controlling; secondly the *developer's* point of view who is usually more interested in a plain task or to-do list.

Based on those four main building blocks we can identify and extract the relevant structures from the process-model to define an appropriate mapping (see Sample Implementations and Experiences).

Audiences for Process-Automation

As stated above, many tools today treat the development in a software project and its management as rather separate disciplines. This certainly has some validity as developers and managers are usually interested in quite different questions. A tool should respect that and target them individually. But instead of having different and potentially incompatible tools for project management and for developers (e.g. loosely coupled tool-set as seen relatively often), we aim at creating a tool infrastructure that treats both viewpoints as different angles on the same subject. The different requirements regarding appropriate tool-support roughly are:

- The *management* usually needs options for planning, controlling and storing documents. Relevant processes are e.g. management, controlling or reporting. Managers are familiar with (project) management and office tools. They are usually interested in cost, schedule and overall project state and trends.
- *Developers* are closer to IDE-tools and don't have a great interest in management issues. They tend to prefer straightforward task lists, compile- or build statistics, code-analysis reports and the likes.

As in [20] we concentrate on these two main use cases to support projects with and without development parts.

Target Environments

As indicated in the roughly sketched use cases, members of project management tend to feel more at home in office tools while developers ideally never have to leave their IDE. Since we want to address distributed development teams, we only look at client-server or web-based tools. To be a candidate for process-tool integration, we require tools to at least have facilities for:

- *Document management*: The target tool should provide document management capabilities including storage, versioning, concurrent working and rights management.
- *Task management*: The tool should provide support for task management. If possible, workflows should be applicable.
- *Collaboration*: The tool should provide capabilities for distributed, collaborative work such as notification mechanisms, message boards and so on. Usually this implies some kind of network-ready applications communicating via intra- or Internet.
- *Workflows*: Since we want to operate process-models, appropriate tools should provide a technical basis for workflows that can be instantiated with processes from the process-model.

In [20] we chose Microsoft SharePoint [11] and Microsoft Team Foundation Server (TFS) [8] as tools to provide sample implementations. Both tools fulfill the requirements listed above. In relation to the use cases, SharePoint was chosen to address management as a web-based platform that provides an easy accessible collaboration platform with basic document, task and workflow management and that seamlessly integrates with the Microsoft Office product family. TFS was used for the development-related use case. In addition to providing the capabilities of SharePoint it provides support for workflow-based work item tracking, modeling, source code control and much more.

Approach

As many tools fulfill the required features stated above, we decided to develop a generic approach. It works as a *model-to-model* transformation [24] engine. Fig. 2 shows the technical architecture of the tool-set that realizes the idea.

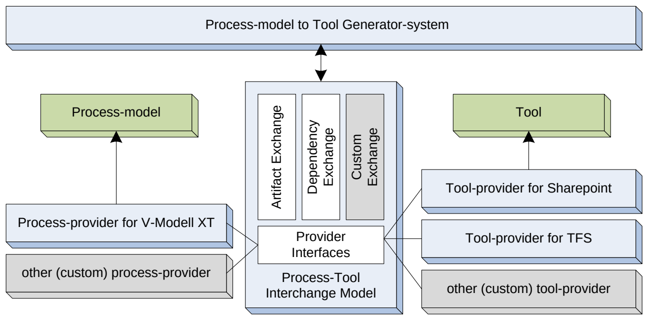

Process-model to Tool Generator-system
Process-model
Artifact Exchange
Dependency Exchange
Custom Exchange
Tool
Tool-provider for Sharepoint
Process-provider for V-Modell XT
Provider Interfaces
Tool-provider for TFS
other (custom) process-provider
Process-Tool Interchange Model
other (custom) tool-provider

Fig. 2 Process/Tool-interchange approach (architecture)

The foundation is a simple intermediate data-model, which is used as translation step between the relevant models. Currently, the model is defined for *artifact* and *dependency* mapping. Based on this core data-model, a set of providers can be implemented. *Process-providers* translate a process model into the intermediate representation which is then accessed by *tool-providers* to create the output. For each input-model a specialized reader is required. Likewise, for every target tool a dedicated writer is needed. The whole translation and generation process is configured and controlled by a generator. The generator is used to connect a concrete process-provider to a concrete tool-provider. This approach is most flexible, as model-readers and writers are independent. So inputs for available target tools can be added, if they are compatible. The other way – new targets for existing readers – works, too. It is also possible to create *multiple outputs* from one input process model. This is useful to have several tools to support the same underlying process instance.

SAMPLE IMPLEMENTATION AND EXPERIENCES

Based on the generator infrastructure we have developed two sample tools that we also presented in [20]. Each tool will be shortly described. A list of features (as far as process integration is concerned) will be given as well as a description of the usage scenarios. Furthermore we provide experiences we have collected from pilot-projects so far.

Target: Microsoft SharePoint 2007

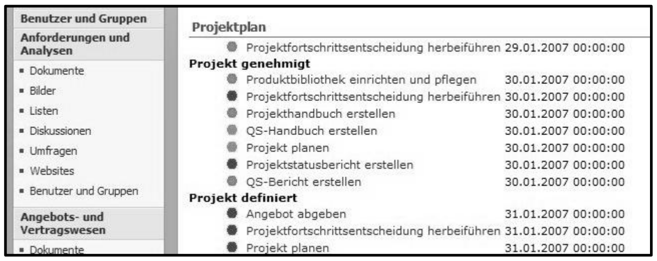
The first reference implementation is provided for Microsoft SharePoint 2007. SharePoint is a web-based collaboration infrastructure with a tight Office-integration. It is sometimes also called Office Server because it offers facilities to create a centralized repository for all types of office documents. Besides document sharing, project team members can share calendars and task-lists, link-lists and much more. Wikis, discussion boards and Blogs are also part of SharePoint.

The provider for SharePoint creates a pre-configured website based on the process contents given by the process provider. Table 1 displays a rough summary of the mapping between the V-Modell XT contents and the resulting elements of SharePoint.

Table 1 Mapping/Translation for SharePoint

V-Modell XT	SharePoint
Products templates	Document templates in a document library
Project disciplines	A document library per discipline including the discipline's process documentation
Activities	Tasks in a SharePoint task list, including process documentation and links to associated products
Initial project schedule	A special task list, including milestones and associated tasks according the schedule

This is just a basic set of mappings. The reason is the audience of this tool. We intended SharePoint to be used in projects without development activities (as can typically be found in the government services). For this scenario, we wanted to provide a simple mechanism to manage project artifacts and some capabilities to coordinate a (distributed) team. Fig. 3 shows a generated sample portal, which is here generated from the German process variant. Because the provider is language independent, an English portal would be generated if an English process were delivered.



Projektplan

●	Projektfortschrittsentscheidung herbeiführen	29.01.2007 00:00:00
Projekt genehmigt		
●	Produktbibliothek einrichten und pflegen	30.01.2007 00:00:00
●	Projektfortschrittsentscheidung herbeiführen	30.01.2007 00:00:00
●	Projekthandbuch erstellen	30.01.2007 00:00:00
●	QS-Handbuch erstellen	30.01.2007 00:00:00
●	Projekt planen	30.01.2007 00:00:00
●	Projektstatusbericht erstellen	30.01.2007 00:00:00
●	QS-Bericht erstellen	30.01.2007 00:00:00
Projekt definiert		
●	Angebot abgeben	31.01.2007 00:00:00
●	Projektfortschrittsentscheidung herbeiführen	31.01.2007 00:00:00
●	Projekt planen	31.01.2007 00:00:00

Fig. 3 A generated SharePoint Portal

To support the management, some ideas from Software Project Control Centers (SPCC) [26] were adopted. As the figure shows, we implemented traffic-light indicators showing the status of work products based on process metadata that was generated into the SharePoint portal. The current implementation is more or less a proof of concept and there is still a long way to go towards a real Project Control Center. Nevertheless this solution is very attractive because of its simplicity.

SharePoint can be manipulated via a rich .NET-based API. Almost every aspect of a SharePoint portal can be changed by programming against that API. With such possibilities, the features one could imagine for process-awareness and process lifetime support are nearly endless. Later in this paper we present some actual work in this field.

Target: Microsoft Team Foundation Server

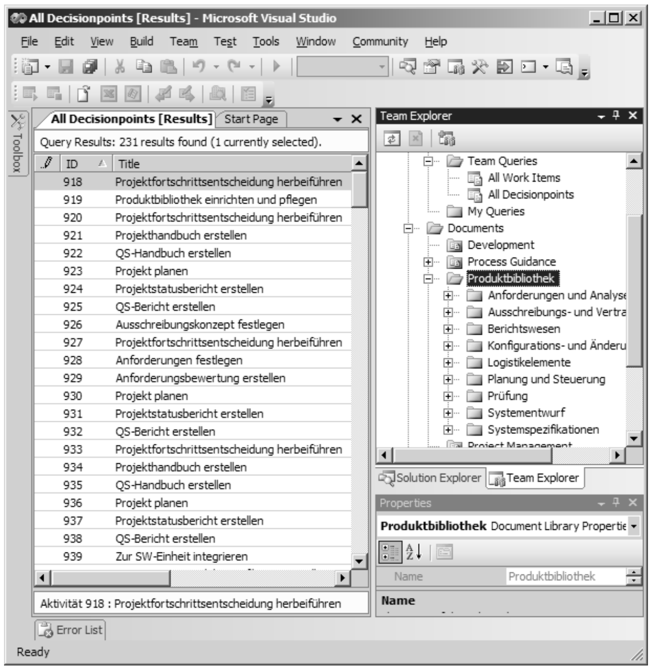
The second implementation from [20] addresses developers in a distributed team. The Team Foundation Server (TFS) is a set of integrated tools including databases, source code control, web portals, report-systems, work item tracking and Microsoft Office integration.

The mapping onto TFS is fundamentally different from the one onto SharePoint. TFS-processes are based on a process template structure [8, 9]. A process template is a simple directory structure containing templates for a SharePoint portal, a set of work item type definitions including a list of defaults, the process guidance, queries and reports.

This means that for TFS the source-process has to be transformed into this static structure. The resulting process template has then to be imported into TFS and can then be instantiated. So while the SharePoint approach directly manipulates a SharePoint portal and can for example create links between interdependent process elements, the TFS approach is more oriented towards prefabrication of a template to be then imported into TFS. In our current solution, the tool-provider just fills that prefabricated meta-process template with contents according to the given process model. Some aspects of the TFS process template remain completely unaffected by generation (e.g. work item type definitions). Table 2 provides a summarized mapping from the V-Modell XT contents to the TFS.

Table 2 Mapping/Translation for Team Foundation Server

V-Modell XT	Team Foundation Server
Product templates	Document templates in a TFS-related SharePoint document library
Products	A special work item type
Activities	A special work item type
Decision gates	A special work item type
Initial project schedule	A collection of work items instantiated from work item types that are associated according the process's definitions.
Process documentation	Copied to special website included in the TFS-related SharePoint instance
Selected, micro processes	Each is provided by a specialized work item type.



Query Results: 231 results found (1 currently selected).

ID	Title
918	Projektfortschrittsentscheidung herbeiführen
919	Produktbibliothek einrichten und pflegen
920	Projektfortschrittsentscheidung herbeiführen
921	Projekthandbuch erstellen
922	QS-Handbuch erstellen
923	Projekt planen
924	Projektstatusbericht erstellen
925	QS-Bericht erstellen
926	Ausschreibungskonzept festlegen
927	Projektfortschrittsentscheidung herbeiführen
928	Anforderungen festlegen
929	Anforderungsbewertung erstellen
930	Projekt planen
931	Projektstatusbericht erstellen
932	QS-Bericht erstellen
933	Projektfortschrittsentscheidung herbeiführen
934	Projekthandbuch erstellen
935	QS-Handbuch erstellen
936	Projekt planen
937	Projektstatusbericht erstellen
938	QS-Bericht erstellen
939	Zur SW-Einheit integrieren

Aktivität 918 : Projektfortschrittsentscheidung herbeiführen

Fig. 4 V-Modell XT integration for TFS (work item view)

Fig. 4 shows an instance of such a generated process template. Here also the German process is shown because of difficulties in the internationalization of the template structure. Since TFS is a server system that can be accessed by several clients (including Office or Eclipse), the clients control how users interact with the server. The server itself works in the background according to the regulations of the instantiated process template – especially if using sophisticated work items, each of which might hold its own micro-process. For detailed information regarding the design of V-Modell-XT-related reference micro-processes see [21] (German).

Experiences and Discussion

Having presented the existing reference implementations of our approach, we now want to give a short discussion. Both providers take a (maybe the same) process instance and provide an appropriate input for the target tools. But the tool-providers actually differ in the implementation technique. The SharePoint provider is directly coded against the SharePoint-API. This opens up all capabilities that SharePoint has. The provider for TFS as it is realized today is limited to the structure of the TFS process templates. This has to be completely filled with all data *before* it is installed into the TFS which is a major drawback compared to the flexibility in the SharePoint solution. Since TFS doesn't provide capabilities for modeling dependencies between process elements declaratively, a lot of semantic of the input process is lost. Currently this problem is addressed by a workaround that stores all relevant metadata of the process elements in the work items themselves. In a second generation-pass done by a separate tool, the metadata is processed and the dependencies between process elements are established.

Nevertheless both approaches fulfill the requirements. The SharePoint solution has the advantage that it is not only almost independent of the input process but also very flexible with regards to the resources used by the portal. So in principle the provider can create and fill portals based on different site- or application templates, which means that an input process can be integrated transparently with existing and living portals with their own corporate identity and so on. It would just create a new site for the project within that portal. In addition to that SharePoint can be augmented by custom workflows, which opens the door for sophisticated user assistance (see next section).

TFS also meets our requirements. Selected reference processes e.g. task- or issue tracking are modeled according the V-Modell XT. Developers are spared of bothersome paperwork but can work with an intelligent work item tracking system. In addition to those reference processes, process elements for measuring the project's state (reports in TFS lingo) are also provided so that a project manager knows about the state of affairs in the development team and can align this information with his schedule or plans. Furthermore the required documentation for the V-Modell XT process is also available to the project team. The TFS implementation is currently under evaluation for practical use in a large government department in Germany.

ONGOING RESEARCH AND FUTURE WORK

With SharePoint and TFS we can provide first usable tool-integration solutions. On the one hand we have the "simple" integration in SharePoint, which enables project managers to easily implement a V-Modell XT compliant project. On the other hand we provide a solution for TFS that takes V-Modell XT concepts onto a concrete development platform. The current toolset is still in fairly early stages and while doing tests and evaluations and when talking to partners in research and in industry, we found many open questions, fields for improvement, desirable features and additional fields of application. A brief overview over the future topics is given in the next few sections.

Application to other Platforms

Currently we provide sample solutions only for the V-Modell XT as process-model and SharePoint and TFS as target tools. It is quite obvious that other combinations would be interesting, too. Currently IBM Jazz with Team Concert [7] is under observation. IBM Jazz is directly comparable to TFS. It also consists of a set of tools that build an infrastructure for distributed development. Foundation for the definition of Jazz-compliant processes is the SPEM-based Rational Method Composer (RMC), which is also known from Eclipse Process Framework (EPF) in a similar form. Currently, we are evaluating the steps necessary to provide a V-Modell XT solution on Jazz using the RMC process definition structure.

We were also asked if our approach would work with SPEM [8] and especially with RUP as process model input. This is also under evaluation. RUP also contains the required sub-models (artifacts, roles etc.) and a formal metamodel. So artifacts, roles, tasks and sub-processes that fulfill the requirements to be implemented with our tools, can be identified. The current tools are meant to be open for such cases. Yet, it has still to be looked into the details of a concrete mapping of these process models (e.g. if the current intermediate data model is powerful enough to adequately represent these processes or if and how it would have to be extended).

Looking at Fig. 2, it is our goal to have a flexible solution where we could just plug in new process model providers or tool providers. The application to another process then would boil down to implementing an appropriate provider that transforms the process model into the intermediate representation understood by the tool providers.

Process-refinement

Besides those technical or practical issues, a lot of methodical research questions are still open. One of the big areas is the whole subject of *process-refinement* [25, 27]. Regarding the V-Modell XT we have to give an answer to the question: *How to provide concrete processes for abstractly described standard contents?*

With the reference implementation we showed a possible embodiment. Standard processes such as *issue and change management* were redesigned using TFS work items. Of course, more organization-specific micro-processes can be identified that should be available through a supporting tool, e.g. *bug tracking*. To accommodate such additions, we are currently working on a concept to transparently integrate methods during the generation. The challenge here is to guarantee the *consistency* of the process. Added elements may change process contents or structure. So a simple merge of contents might affect consistency if the newly added elements are not known in the process. If for example an additional process for bug tracking is established in the project, but bug tracking is not a content of the original process or the original process is not made aware of the addition, this could lead to inconsistent process-/project-models and potentially endangers process certification, e.g. related to a CMMI-level. The *tool-based method tailoring* (Fig. 5) provides a concept which allows a "late-binding" tailoring respecting the process's consistency.

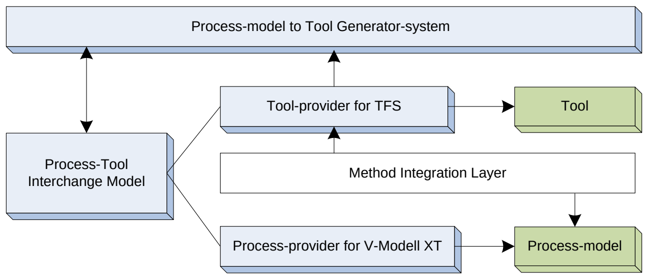

Process-model to Tool Generator-system
Tool-provider for TFS
Tool
Process-Tool Interchange Model
Method Integration Layer
Process-provider for V-Modell XT
Process-model

Fig. 5 Process-refinement by method tailoring

The V-Modell XT for example defines explicit hot spots for extension so that extension-points are computable during the whole life cycle. Method tailoring includes the provision of an additional *method integration layer*. This layer relies on both the process model and the target tool. The layer itself contains *method packages* (e.g. work items in the TFS setting) that should be made available in the target tool. Method packages are taken by the tool-provider and, depending on the target tool, converted for example to TFS work items and then injected at the correct location in the target system. Furthermore, the process model is made aware of the methodical addition, for example by attaching the method to V-Modell XT extension points. This could go as far as to inject additional documentation at appropriate locations in the overall process documentation. As far as we can see right now, this can be done declaratively as well as imperatively.

Sophisticated User-Assistance

Another point of interest is the provision of runtime support during a project's runtime. Our current concepts aim at minimal impact on the tools used for process integration – meaning that no additional tools or add-ins must be installed on the machine running the tool to be addressed. As a result, for example when using TFS, we are limited to purely declarative process integration. As mentioned before, this means that for TFS we can provide only a subset of the features of the V-Modell XT as we are limited by the TFS process template structure. Additional functionality would require programmatic solutions that would have direct impact on the particular tool.

Nevertheless such functionality might desirable. For SharePoint we are currently evaluating a concept to assist users in the creation of work products. The V-Modell XT contains a concept called *creational dependencies* that says that if a certain product was created, some other products have to be created, too. If for example an architect finished a system specification, the creational dependencies say that he now may create an architecture document or a set of specifications representing the overall system's decomposition. The broader question behind this is: *Given the artifact I am just working on, what steps are possible, sensible or even required next?* As the creational dependencies in the V-Modell XT span a dependency graph between artifacts, the dependency structure can be analyzed at runtime (Fig. 6) using simple graph-algorithms.

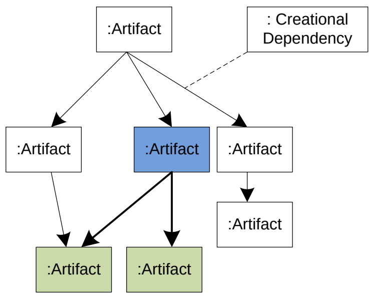


Fig. 6 Calculating next artifacts using a creation-tree

Starting at any product/artifact stored in the document library, possible paths and next steps can be calculated. Given a starting artifact, the user is thus offered with a list of artifacts that the process model thinks should be created, too. If the user selects an artifact for creation, it is stored into the appropriate SharePoint document library and a new task associated to it is created. The task will be assigned to a role, which is responsible for the new artifact. SharePoint users in that role will be notified automatically.

For TFS comparable add-ons are desirable. If for example a development project needs an additional iteration, a lot of process-related artifacts have to be created and correctly connected to each other. A solution for this problem would be the extension of the TFS project organization. At the moment, such an extension has to be done manually by the TFS team project manager. But one could imagine a system that automatically creates the necessary artifacts as prescribed by the underlying process, adjusts the project plan accordingly and much more.

Product Data Modeling

The fourth area that we are looking into is the subject of Product Data Modeling (PDM) [22, 23]. To be applicable in as many project scenarios as possible, the V-Modell XT specifies product contents only on a relatively abstract level at all. At the moment, the SharePoint solution for example treats work products as black boxes (actually as RTF documents) that are annotated with some additional process-relevant metadata. While certain products, such as specifications, are so free in their structure and contents, we believe that certain product contents can be modeled explicitly in the target tool. Candidates for such an explicit modeling are items in a bug list, status reports, testing protocols, system architectures and so on. Such *structured products* could be edited in the target tool. We therefore were exploring the possibilities of integrating Microsoft InfoPath forms into SharePoint. We can see several benefits from the tool being aware of product contents:

- The tool could check user input (for example into a form) for consistency.
- Imagining a product such as an issue or project risk list, the tool could provide reports based on the contents in the list. It could for example display statistics on how many issues are unresolved or if a risk does not yet have a mitigating strategy.
- Again imagining an issue list, a particular issue could be linked directly to the artifact that it relates to. This would allow the tool for example to display all issues related to the work product currently looked at.

A drawback of explicit product content modeling is that it possibly limits expressiveness. We will have to be careful, which product content to model and which product to better leave unspecified to not constrain users in how they have to finish the product.

In our TFS implementation we are currently looking at replacing the generic system (under development) model as defined by the V-Modell XT by the "real" system model as it is implemented in source code. We have yet to investigate how the implemented system model can be related to elements prescribed by the process model.

CONCLUSION

In this paper we gave a short introduction to our meta-model-based tool/process-integration. We use the power of existing formal process-models, using the German V-Modell XT as sample, to map them onto actual collaboration and development environments, thus creating process-aware tools for distributed teams.

First we presented the approach and first implementations. We specifically discussed the existing tool-providers for Microsoft SharePoint and Team Foundation Server, presented first experiences and listed pros and cons. Most of the drawbacks and limitations had to do with the declarative approach we used in the TFS realization. We motivated future work and ideas, not at least as asked for by industrial practice. We outlined additions during the generator stage to include additional method components as well as additions for the addressed tools to improve user assistance at runtime. We gave a rough impression of our current research, which mostly deals with said additions and the inclusion of other platforms, e.g. IBM Jazz.

With our approach we bring together two worlds: We use formal, integrated process-models as input for a generator that produces whole working-ready environments for distributed teams using collaboration and development tools. The approach is aimed at supporting teams in their familiar environments using a structured process. The advantages are:

- Project members, including management and development roles, still work with their familiar tools.
- Processes can be transparently established using tools already in use in the team. This, as we hope, has a positive impact on the process's acceptance.

- Process elements that are appropriate for automation can be supported by the tool. Bothersome work such as reporting etc. hopefully can be (partially) automated.
- Using a platform that supports distributed and collaborative work for process enactment provides a way to coordinate a team at different locations. The project is distributed but all information, data and artifacts are available at any place.
- If a customer requires a particular process, it can be implemented with a minimum of additional effort.

Of course, many research questions remain open. When trying to integrate development environments and process-models, we observe a gap between the *system-model*, which is implemented in source code using a development tool (and some accompanying artifacts, e.g. design artifacts or documentation), and the *project-model*, which is usually instantiated in a project management tool. We often found that those two models to not coincide. The management often has an understanding of the system that only in parts reflects the implemented reality. It would be very interesting to further examine if and how both views can be brought into harmony. In [31] an approach is described that puts the system model first and derives project planning information from that model. This we want to investigate further, hopefully making the tools more aware of the project and its contents and thus being able to provide richer information and assistance.

REFERENCES

1. Ahmed Boulila, N., *A Framework for Distributed Collaborative Software Design Meeting*. PhD Thesis, Technische Universität München, 2005.
2. Allen, T. et al., *Managing the Flow of Technology: Technology Transfer and the Dissemination of Technological Information within the R&D Organization*, MIT Press, Cambridge, MA, USA, 1984.
3. Ameluxen, C., Königs, A. Rötschke, T., Schürr, A., MOFLON: *A Standard-Compliant Metamodeling Framework with Graph Transformations*. In Model Driven Architecture – Foundations and Applications: Second European Conference, Springer, 2006.
4. Becker, S. M., Haase, T., Westfechtel, B., *Model-based a-posteriori integration of engineering tools for incremental development processes*. In Software and System Modeling 4(2), 2005.
5. Becker, S. M., Herold, S., Lohmann, S., Westfechtel, B., *A graph-based algorithm for consistency maintenance in incremental and interactive integration tools*. In Software and System Modeling, 6(3), 2007.
6. Raghvinder, S., Bass, M., Mullick, N., Paulish, D. J., Kazmeier, J., *Global Software Development Handbook (Auerbach Series on Applied Software Engineering)*, Auerbach Publications, Boston, MA, USA, 2006.
7. IBM Jazz overview, http://www.ibm.com/rational/jazz/.
8. Microsoft Corporation, *Team Development with Visual Studio Team Foundation Server*, Microsoft Press, 2007.
9. Guckenheimer, S., Perez, J. J., *Software Engineering with Visual Studio Team System*, Addison Wesley, 2005.
10. Heller, M., Schleicher, A., Westfechtel, B., *Graph-Based Specification of a Management System for Evolving Development Processes*. AGTIVE 2003, 2003.
11. Murphy, A., Perran, S., *Beginning SharePoint 2007: Building Team Solutions with MOSS 2007 (Programmer to Programmer)*, Wrox Press, 2007.
12. Koordinierungs- und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung, *V-Modell XT Online Portal*, http://www.v-modell-xt.de/.
13. Apache Software Foundation, *Apache License*, Version 2.0, January 2004, http://www.apache.org/licenses/LICENSE-2.0.
14. Object Management Group, *Software Process Engineering Metamodel*, http://www.omg.org/technology/ documents/formal/spem.htm.
15. Kazman, R., Kruchten, P., Nord, R. L., Tomayko, J. E., *Integrating Software-Architecture-Centric Methods into the Rational Unified Process*. CMU/SEI-2004-TR-011, Software Engineering Institute, 2004.
16. Königs, A., Schürr, A., *Tool Integration with Triple Graph Grammars – A Survey*. In Proceedings of the SegraVis School on Foundations of Visual Modelling Techniques, Elsevier, 2006.
17. Kruchten, P., *The Rational Unified Process: An Introduction (2nd edition)*, ISBN-13: 978-0201707106, Addison-Wesley Professional, USA, 2000.
18. Eclipse Foundation, *Eclipse Process Framework (EPF)*, http://www.eclipse.org/epf/
19. Kuhrmann, M., Niebuhr, D., Rausch, A., *Application of the V-Modell XT - Report from A Pilot Project*, In Unifying the Software Process Spectrum, International Software Process Workshop (SPW 2005), 2005.
20. Kuhrmann, M., Kalus, G., Diernhofer, N., *Generating Tool-based Process-Environments from formal Process Model Descriptions - Concepts, Experiences and Samples*, In proc. of the IASTED International Conference on Software Engineering (SE 2008), ACTA Press, 2008.
21. Kuhrmann, M., Kalus, G., *Werkzeugspezifisches Tailoring für das V-Modell XT*. Technical Report, TUM-I0804, Technische Universität München, 2008.
22. Karcher A., Wirtz J.: *Requirement Engineering for PDM/EDM Implementation in Virtual Enterprises*. In: International EDM/PDM Symposium '98, 1998.
23. Bender K., Karcher A., Bindbeutel K., Glander G.. *Framework Technology for Tool Integration in Integrated Product Development*. International Conference on Manufacturing Automation, 1997.

24. Frankes, D. S., *Model Driven Architecture - Applying MDA to Enterprise Computing*, Wiley & Sons, 2003.

25. Münch, J., *Transformation-based Creation of Custom-tailored Software Process Models*. In Proceedings of the 5th International Workshop on Software Process Simulation and Modeling (ProSim 2004), 2004.

26. Münch, J., Heidrich, J., *Software project control centers: concepts and approaches*, In The Journal of Systems and Software, 2004.

27. Osterweil, L. J.: *Unifying Microprocess and Macroprocess Research*. In proc. of International Software Process Workshop (SPW 2005), 2005.

28. Rausch, A., Bartelt, C., Ternité, T., Kuhrmann, M., *The V-Modell XT Applied – Model-Driven and Document-Centric Development*. In: 3rd World Congress for Software Quality, 2005.

29. Rombach, D., *Integrated Software Process and Product Lines*. In proc. of International Software Process Workshop (SPW 2005), 2005.

30. Watts, H., *Three Process Perspectives: Organization, Teams, and People*. Annals of Software Engineering, Volume 14, Kluwer Academic Publishers, 2002.

31. Beneken, G. H., *Logische Architekturen – Eine Theorie der Strukturen und ihre Anwendung in Dokumentation und Projektmanagement*, PhD Thesis, Technische Universität München, 2008.