



and INTSPEI P-Modeling Framework. We also discuss related work – the customizations of MSF made by other authors.

Microsoft Solutions Framework is an effective software solutions development approach which is based on Microsoft experience. MSF was first introduced in 1994 as a set of best practices from Microsoft's product development efforts, as well as the activities of Microsoft Consulting Services. After introducing MSF, Microsoft gathered feedback from their software development groups, partners, and customers throughout the world. As a result, MSF was improved and evolved, and after publishing several versions, Microsoft introduced MSF customizations for CMMI and Agile processes [1]. The current version of MSF is version 4.1; we integrated P-Modeling with this version.

The MSF Agile (short name for “MSF for Agile Software Development”) process is a version of MSF intended for teams that adopt agile development principles [1]. It supports scenario-driven, context-based software development. MSF Agile can also be used as a first step in adopting a more formal MSF CMMI.

The MSF web-site [1] lists a number of process templates intended to customize or replace the MSF Agile or MSF CMMI processes. However, the process templates presented on the MSF web-site [1] either replace MSF guidance with entirely new methodology, such as Conchango SCRUM, Cognizant FDD or Ivar Jacobson EssUP, or use unmodified MSF guidance and instead focus on more technical aspects, such as BrightWork PMPPoint, providing a SharePoint portal for project managers based on MSF. We have not found any examples of more in-depth integration, combining elements of both original MSF methodology and a custom process.

INTSPEI P-Modeling Framework originates from “The Babel Experiment” designed by Vladimir L. Pavlov in 2001 as a training program [7]. It is based on two principles: Reverse Semantic Traceability and Speechless Modeling.

Reverse Semantic Traceability (RST) is a quality control method that allows testing outputs of every “translation” step. Before proceeding to the next phase, the current outputs have to be restored and compared to the original artifacts. If they are semantically different, then the translation step has to be repeated more precisely to eliminate this ambiguous understanding. This reverse step and the evaluation of the degree of semantic correctness must be performed by human.

The key word in the name of this method is “Semantic” because the original and restored versions of an artifact are to be compared semantically, with a focus on the “meaning” of this artifact, not on particular notions used in it.

Speechless Modeling requires that UML is the only language allowed for communication while working on the project design, and any verbal or written communication involving natural languages is forbidden. This constraint provides the following benefits [9]:

- It stimulates creativity of designers and forces them to stay focused on the task;
- It forces designers to specify explicitly all assumptions;
- UML is no longer treated as a superfluous burden; instead, designers start to care about readability and the quality of their models.

P-Modeling Framework was initially designed as an add-on to existing methodologies. It is not a new process, describing an entire software development lifecycle. Instead, it provides powerful quality control and modeling techniques that can be introduced into any existing process. P-Modeling Framework can be valuable for both agile and more formal processes [9].

INTSPEI P-Modeling Framework is not discussed here in detail. More detailed discussion of this methodology can be found in [7-10].

### 3. Points of integration

This section describes the major points of the integration of P-Modeling Framework into MSF Agile. We describe elements of MSF that were modified to include content specific to P-Modeling. In particular, we will focus on the exact locations in the MSF lifecycle where Reverse Semantic Traceability and Speechless Modeling could be used.

To select possible points of integration, we used a process similar to MSF itself. We performed conceptual design, logical design and then physical design. First, we examined the high-level description of the MSF Agile lifecycle to identify the possible options for integration (conceptual design). For each option, we then elaborated the details of integration (logical design). Finally, we identified how we would represent new content in terms of the MSF metamodel (physical design).

An important note is that conceptual design was performed for MSF version 3. The third version of MSF did not contain the detailed process guidance that appeared in MSF version 4. Therefore, the high-level options we found represented all possibilities of integration. With the release of MSF version 4, we decided not to perform another high-level step. Instead, we tried to elaborate upon the options found earlier using MSF version 4.

#### 3.1. High-level integration (Conceptual Design)

In this first stage, we considered MSF Agile Tracks (which were called Phases in MSF v3) and Disciplines. They provide a high-level view of the project lifecycle from different perspectives: Tracks describe the evolution of the project, while Disciplines span across entire project, identifying specific areas of concern. Another possible approach for the high-level overview of the lifecycle was to focus on Roles. However, P-Modeling practices usually involve multiple roles, and roles in a project are not as important for P-Modeling as expertise.

Therefore, we decided to limit ourselves to Tracks and Disciplines.

The next subsections present points of integration that we found in each track or discipline, together with a brief description of the track/discipline.

**Risk Management Discipline** is aimed at controlling risks – uncertainties that can affect project success.

In addition to regular work with risk, RST sessions equip the team with a dedicated validation means, checking that risk planning matches the risk which it addresses. In particular, during an RST session, risk descriptions are restored from contingency and/or mitigation plans, and then discrepancy analysis is performed.

**Envision Track.** This track starts the project. The goals are to form the core team and make sure that the staff possesses the required skills, understanding the scope of the project, as well as outlining the high level architecture of the system, baselining the major risks, and underlining the plan of product development. Thus, the deliverables are the project structure, risk assessment and the vision/scope documents.

The Vision Document is the crucial deliverable of the Envisioning Phase, aimed at establishing the future of the project. Two major parts of the document are the conceptual description of the prospective product and the set of approaches (i.e. testing, coding, deployment, etc) to its development. RST can be employed to ensure approaches really serve the project goals.

The team baselines conceptual design during the Envision Track, and, for a few reasons, this is the best time to conduct a P-Modeling Session. First, speechless mode will help the team to create the highest quality model. Second, the RST session will validate and improve the created model. Last, this one day event will enable people to start working together as a solid team and will help to detect potential true leaders within the team. The ultimate result of the session is an established agreement within the group on how the future solution will work and how this result will be achieved.

**Plan Track.** The purpose of planning is to elaborate upon and document the work required for creating the system and to provide internal and external stakeholders with comprehensive information on the functionality to be implemented. The phase deliverables are: Functional Specification, Risk Management Documentation, and the Master Project Plan and Schedule.

Examples of RST sessions during planning are: reverse mapping the Master Project Plan to the Vision Document, project schedules to project plans, Conceptual Design to Requirements, Logical Design to Conceptual Design and Physical Design to Logical Design, and the various elements of design to requirements that caused them, etc.

**Build and Stabilize Tracks.** During the Build Track, the team implements everything that was planned during the previous phase, including documentation and related infrastructure. The Stabilize Track follows the Build track and is aimed at addressing the issues that were introduced during the development of the solution. The team focuses

on finding and fixing bugs and prepares the system for release. The deliverables of the Stabilizing Track include: a stable release and release notes, project documentation and test results.

The primary mission of INTSPEI P-Modeling Framework during development and stabilization is to verify if the implementation meets the requirements elaborated in the planning phase. The proposed quality control activities include restoring the elements of the model from the code modules in an RST session, restoring changes in the code caused by a bug fix into the bug description, restoring elements of the model that describe certain functionality from test cases written to test this functionality and others.

**Deploy Track.** During the deployment, the project team transitions the ownership of the product to the customer(s). The INTSPEI P-Modeling Framework usage is not as intensive during this period as compared to previous phases. However, the RST sessions still contribute to project quality. For example, having a deployment plan, the team restores the deployment requirements and, thus, can assess the extent to which the plan for deployment at a customer's site implements the requirements for this particular site.

### 3.2. Logical design

After the high-level options for integration were identified, we evaluated the following options to obtain the final list of P-Modeling activities to be included into the MSF Agile lifecycle.

For risk management, we wanted to add RST for risks; however, closer examination of the MSF Agile templates revealed that the Risk work item does not contain fields for contingency and mitigation strategies. Such strategies are supposed to be recorded in the same field with the description. This means that, in order to perform a Reverse Semantic Traceability session, the risk description fields should be processed to split them into the actual description (which will be restored during RST) and the contingency/mitigation plans (which will be the inputs to RST). Based on these observations, we decided not to include a specific RST action for risks.

When we evaluated the changes in tracks from MSF v3 to MSF Agile v4, we found that the number of work products was reduced and many work products were simplified. In particular, we did not include RST for Vision (Envision Track), because it no longer contains a second part describing the set of development approaches.

For the Plan Track, we decided to implement RST only for the Scenario (instead of Requirements) and for the Architecture (instead of multiple options of RST for designs of different levels). As MSF Agile discourages unnecessary bureaucracy, we decided not to include RST sessions for plans and schedules. While these work products are important to overall project success, they are not primary objectives of the project. We also included a Speechless Session to create the initial version of the architecture.

For the Build Track, we included RST for implemented code (restoring either the task or bug description from code). We also included RST for test cases, restoring Scenarios or Quality of Service requirements. Notice that instead of design (such as UML models), we decided to restore textual descriptions, because this practice simplifies RST sessions.

In total, we identified 7 RST tasks that should be included in MSF Agile:

- Perform RST for Scenario;
- Perform RST for Solution Architecture;
- Perform RST for Development Task Implementation;
- Perform RST for Database Task Implementation;
- Perform RST for Bug Fix;
- Perform RST for Scenario Test Cases;
- Perform RST for Quality of Service Requirement Test Cases.

Also one task for a Speechless Session was included (Conduct P-Modeling Session).

An interesting observation is that all P-Modeling tasks are part of the Build Track in MSF v4. This happened because the MSF v3 content was restructured, so that most work products are now created in the Build Track.

### 3.3. Physical design

When the P-Modeling tasks to be included in the MSF Agile lifecycle were identified, we needed to decide how these tasks should be represented in the MSF metamodel. Two possible options were to represent them either as activities or as workstreams. The difference between these options is that the workstream in MSF consists of activities.

The representation of P-Modeling tasks as workstreams has the advantage of detailed representation of multiple steps required for each task. The Reverse Semantic Traceability session consists of the following steps: Preparation, Reverse Engineering, Expert Assessment and Making a Decision [10]. Each of these steps can be represented by an activity that has its description, input and output work products, can be tracked using work items, etc. For example, the output from the Reverse Engineering step is a restored version of a work product, and it is the input to the Expert Assessment. The output from the Expert Assessment is a comparison of the original and restored versions of the work product, and the list of identified differences.

However, the limitation of MSF workstreams is that they cannot be nested. Therefore, it is not possible to indicate that Reverse Semantic Traceability sessions are valuable not by themselves, but as part of other activities. Because of these considerations we decided to represent RST tasks with activities that are incorporated into corresponding workstreams. For example, we included the activity “Perform RST for Scenario” as a part of the “Create a Scenario” workstream.

Another consequence of the decision to represent RST sessions with activities instead of workstreams is that these activities are scattered across different parts of the MSF lifecycle. To provide a single viewpoint of these activities, we created a discipline “Traceability Management”. This discipline groups together all activities related to Reverse Semantic Traceability.

### 4. New practice: plan RST activities

During the integration of P-Modeling Framework with MSF Agile, we noticed the need for a new practice in P-Modeling Framework itself. Most of the RST activities that we incorporated into the MSF lifecycle could possibly be applied to a large number of project artifacts. Examples include parts of code, test cases and scenarios. During a typical MSF project, there will be a large number of such artifacts, and performing RST for each of them can create overhead and should be well justified if chosen – e.g. for medical software where possible information loss is very critical.

It became apparent that we needed a method for prioritizing work products, so that RST is performed only for a few selected work products. We propose the technique for prioritizing artifacts based on their importance for project success and level of quality control applied to them.

Prioritizing artifacts is performed during the iteration planning stage, which is represented by the “Plan RST Activities” task. First of all, the project manager should create a list of all artifacts the project team will have during the project. They could be presented as a tree with dependencies and relationships. Artifacts can be present in one instance (like a Vision document) or in multiple instances (like risks or bugs). This list can be changed later during the project, but the reasoning behind the decisions about RST activities will be the same.

Importance \ Level of quality control	1 – Crucial	2 – High	3 – Medium	4 – Low
1 – Low	1 - RST is required	2 - RST is required	3 - RST is recommended	4 - RST is recommended
2 - Medium	2 - RST is required	4 - RST is recommended	6 - RST is recommended	8 - RST is recommended
3 - Sufficient	3 - RST is recommended	6 - RST is recommended	9 - RST is recommended	12 - RST is not required
4 - Excellent	4 - RST is recommended	8 - RST is recommended	12 - RST is not required	16 - RST is not required

Fig.1. RST Rank Table

The second step is to analyze deliverable importance and the level of quality control for each of the project artifacts. The importance of the document is the degree of artifact impact to project success and to the quality of the final product, measured by a 4-grade scale: Crucial, High, Medium and Low. The level of quality control is a measure that defines the amount of verification and validation activities applied to the artifact, and the probability of miscommunication during artifact creation. It is also measured on a 4-grade scale: Low, Medium, Sufficient, and Excellent.

To complete the planning, the project manager evaluates the list of all project deliverables and associates each delivery with two values: importance and level of quality control. Multiplying these two values gives the RST rank. (A similar technique is used in MSF for calculating risk exposure [1]). The decision to perform RST for the artifact is based on this rank – the lower the rank, the more RST is recommended. (See Fig. 1)

## 5. Results of integration

After all of the decisions described in this paper were implemented, we obtained a version of MSF Agile guidance that included P-Modeling Framework. The development effort took about 2 months. Fig. 2 shows the new discipline, Traceability Management, as well as all P-Modeling Activities in their workstreams.

We introduced 9 new activities into MSF, bringing the total number of activities to 96. Therefore, the percentage of RST activities is 9%. The number of new activities shows the amount of new content added to the lifecycle – in this case, it is not very significant.

P-Modeling tasks were added to 8 workstreams out of 17, or to 47% of all workstreams. The number of modified workstreams suggests the fraction of the lifecycle that was affected by the new content. For P-Modeling activities, this number is significantly greater than the percentage of added tasks.

Some companies using MSF have expressed their interest in P-Modeling Framework integrated with MSF Agile. They have started a number of projects based on the integrated process. While we still have not evaluated the effectiveness of our process based on the results of these projects, the initial feedback suggests that our goal was accomplished. Development teams using MSF are now able to employ P-Modeling techniques as a natural part of their development process.

The integrated process can be downloaded from <http://www.intspei.com/Products/PMFramework.aspx>.

## 6. Conclusion

In this paper, we described our experience with integrating our P-Modeling Framework with MSF for Agile Software Development. We showed that the available guidance on process customization covers mostly technical aspects, leaving out essential questions,

such as identifying possible points of integration. The process of selecting an approach for integration has been described in detail. As part of making these decisions, a new technique for our add-on has been identified and developed. Finally, we presented the results of the integration.

An important discovery that we made is an appearance of a new technique, namely prioritizing artifacts for RST. Of special interest is the fact that we recognized the need for defining this technique during integration. This example shows how two different methodologies can benefit from integration. They not only enrich each other, but also produce new techniques not present in either of the original methodologies.

Possible directions of future research include integrating P-Modeling Framework with other processes, including MSF CMMI, RUP and others. We believe that such integration will lead to new insights about the usage of P-Modeling. Another topic of research is comparing integration possibilities when using different tools and methodologies.

## Acknowledgements

We would like to thank all people who reviewed different versions of this article, especially Bekah Sondregger, Viktor Sergienko and Petro Protsyk.

## References

- [1] Microsoft Solutions Framework <http://www.microsoft.com/msf>
- [2] Ph. Kruchten, *The Rational Unified Process: an introduction* (Addison-Wesley, 2003).
- [3] K. Beck, *Extreme Programming explained: embrace change* (Addison-Wesley, 2000).
- [4] I. Jacobson, P. Wei Ng, I. Spence, Enough process - let's do practices, *Journal of Object Technology*, 6(6), 2007, 41-66.
- [5] Rational Method Composer <http://www.ibm.com/software/awdtools/rmc>
- [6] VSTS Process Guidance Generator <http://www.codeplex.com/process/Release/ProjectReleases.aspx?ReleaseId=5626>
- [7] V. Pavlov, A. Yatsenko, Using pantomime in teaching OOA&OOD with UML, *Proc. 18<sup>th</sup> IEEE Conference on Software Engineering Education and Training*, Ottawa, Canada, 2005, 77-84.
- [8] V.L. Pavlov, N. I. Boyko, A. V. Babich, First experience of using INTSPEI P-Modeling framework in software development projects, *Problems In Programming*, (2), 2007, 68-75.
- [9] V. L. Pavlov, S. Busygin, N. Boyko, A. Babich, Is there still a room for programmers' productivity improvement?, *Proc. 5<sup>th</sup> East-West Design and Test Symposium (EWDTS'07)*, Yerevan, Armenia, 2007, 146-151.
- [10] INTSPEI P-Modeling Framework Whitepaper, INTSPEI, <http://www.intspei.com>

## Workstreams used in 'Traceability Management' (Disciplines)

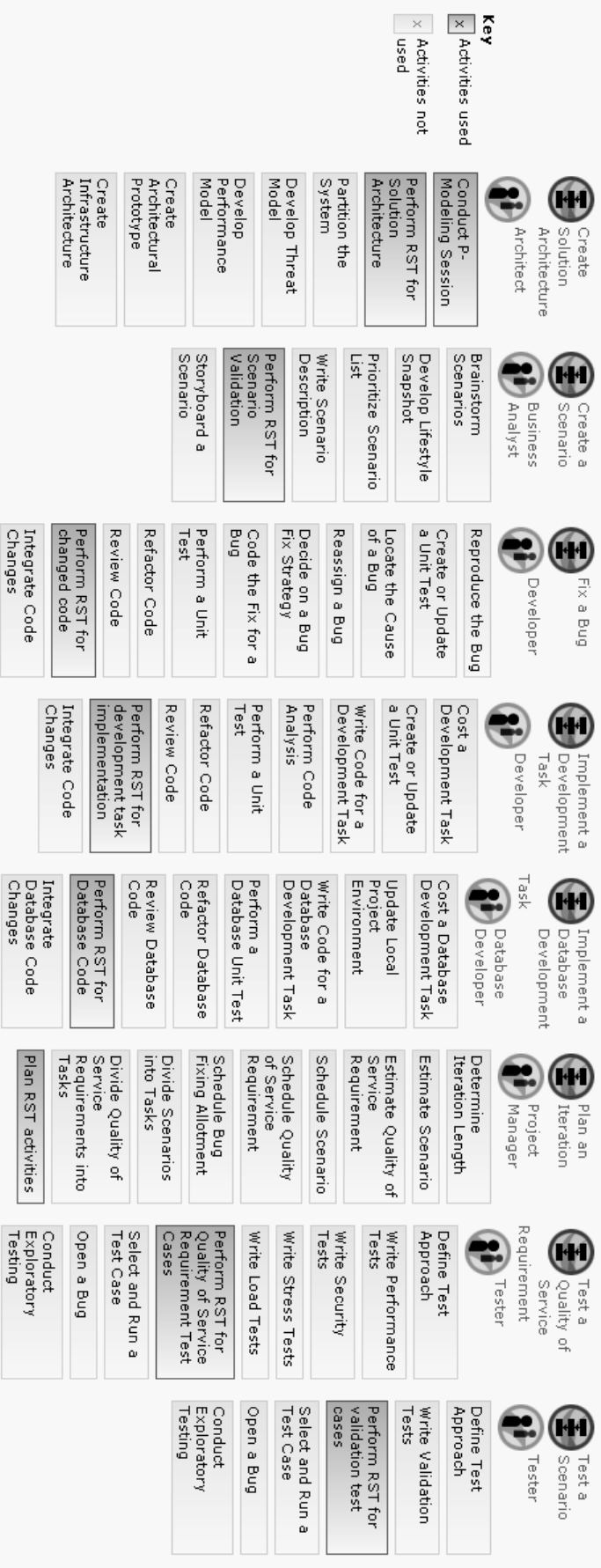


Fig. 2. P-Modeling activities in MSF workstreams