Abstract

The aim of this article is to present the project framework for constructing a Software Process Simulation Modeling (SPSM) system. SPSM systems can be used as a virtual environment for the selection of methods and tools of project management in IT support organizations. The constructed system simulates the Scrum methodology, including the management processes and the project roles. For the implementation of Scrum processes, the Scrum ontology is proposed and for the competences of the roles of project team members, a fuzzy-logic representation. As a result we present the hybrid fuzzy-ontological system. The framework of the design processes proposed in the article was verified on the basis of real courses of project management processes in a large IT company.

Keywords: SPSM, Software Process Simulation Modelling, Ontology, Fuzzy Logics, Scrum;

1. Introduction

For the planning and control of manufacturing processes in IT projects, Software Process Simulation Modeling (SPSM) environments are used. The first works on the subject appeared relatively long ago, and covered the importance of these environments for software development teams. The first publication in this field appeared in 1998 (after the ProSim4 workshops). It was Kellner and Madachy who, in their work, presented an analysis of the
construction and meaning of these environments\textsuperscript{11}. They suggested treating the development environments as a group of processes - a logically connected structure of people, technologies and practices organized to perform a task. They treated SPSM as an abstraction of a real or conceptual development system.

The system proposed in the article is treated as a virtual project environment to support the work of those who manage project teams in the planning and monitoring of projects. It relies on the agent-based design paradigm and uses intelligent inference mechanisms to support its functionalities. It can be applied in the work of project teams (e.g. in an IT support organization) and implemented on the basis of any operating system, as well as IT infrastructure\textsuperscript{14}. The main application of this system is seen in the planning of IT projects at the stages of constructing the team, defining tasks for the team and selecting methods and tools for the development and management of projects. This approach helps the project manager to plan the budget, schedule and scope of the project for any client requirements.

The process of building a project framework of SPSM was divided into four stages. The first stage involved defining the system variables to determine the project framework and the possibility of applying ontologies and fuzzy modeling in the developed model. In the second stage, the Scrum\textsuperscript{2} glossary was built and ontologies were developed for the needs of the system. In the third stage, competence patterns were used for the construction of a fuzzy inference model for the selection of project team members.

2. Defining the variables of the SPSM model to determine the project framework and implementation issues - Stage I of the research

Figure 1 shows the general SPSM model simulating the Scrum method. It includes the classic process approach to the description of the organization of an IT service provider. The input variables were extracted, describing: the size of the product backlog as the number of points in the user history and the complexity of the project on the basis of the definition of the tasks necessary to recognize items of the product backlog as finished. On the basis of these specific variables the proposed system predicts: the speed of software development by a team, the risk of the project, the workload of the team members and the applicability of the tools used.

For processing the input variables, it becomes necessary to apply initial processing for defining agents and the object-oriented mechanisms of their analysis. Then the proposed model creates conditions for the analysis of one or more Scrum iterations/sprints. In addition, thanks to the initial processing the status of the project at the output of the dynamic model\textsuperscript{16} becomes possible to determine. It is represented by the current contents of the product backlog. This content can be modified or updated based on the actual status of the software development process. The
analysis of the project at the level of an individual sprint allows for the simulation of an IT project which is in progress. It also allows for its course to be corrected.

For the implementation of the model (along with the Scrum process) the JADE environment for building agent systems was used\(^1\)\(^{10,16}\) (a high-level SPSM architecture was based on this environment). The choice of an agent-based environment resulted from the need to implement the behavior of the project team members\(^5\). Then the resources of the project environment are implemented by objects of the multi-agent system environment. It also resulted from the need to implement project management processes. The implementation of a process is carried out by single, atomic tasks executed by agents. These tasks will be implemented in the agent code and they can only be configured and combined into larger entities with the use of the process definition. An example of such an action is communication with the owner of the product in order to clarify issues related to the implementation of the sprint backlog items or the development of the sprint backlog items.

In the Scrum method, the duration of a sprint and events is determined during the project. The execution of such different processes requires the use of a configurable process manager which has several modes of operation implemented. Thus, the definition of the development process includes a configuration of the process manager, which defines roles and assigns tasks to them. In the different phases of the sprint, the set of tasks for which each person with a role is responsible varies. Tasks comprise of smaller units - operations, the set of which is described by a definition of completion common to all product backlog items and by determining which previous work must be completed before the work on the backlog item can be considered to be completed. For the task of the "implementation of a sprint backlog item" such work is, for example, writing a code.

![Diagram of Development Process Definition](image)

**Fig. 2. A fragment of the development process definition**

Here, the definition of the development process has been implemented as a domain ontology\(^13\), in which a general description of the development process is defined consisting of concepts and relations between them\(^9\). In contrast, the corresponding implementations of processes existing in organizations have been defined in a separate ontology - the configuration of the definition of the development process.

In the course of constructing the SPSM, an issue that was also addressed was the lack of differentiation between the time spent on tasks and their type. In the SPSM, there was no possibility of identifying the characteristics and competences of the project team members. Personality traits influence the efficiency of communication between individuals, and the level of competence affects the duration of tasks and the need for communication with other team members to supplement domain or technological knowledge. The level of competence is described linguistically, while the duration of the atomic tasks of the system is determined by the level of competence of the individuals performing tasks, the level of competence needed for accomplishing each task and using the given tool. Here is a scenario of the behavior of agents representing developers, which takes into account the competence levels
and personality traits:
1) During the daily Scrum, developer A reports a problem with a task due to lack of competence in a certain technology.
2) Developers B and C with the appropriate level of competence offer their help.
3) On the basis of their personality traits, developer A chooses developer B with whom he has a better contact and communicates with them for help. The communication time depends on the combination of the personality traits of both developers.
4) The task run time is determined on the basis of the elevated level of competence of developer A.

3. The construction of the Scrum glossary of terms and the development of an ontology for SPSM

The next stage of the construction of a project framework was to define the semantic framework, defining the roles of actors taking part in it, the hierarchies of responsibility and the internal processes. To complete these tasks, it became necessary to adequately represent the knowledge of the above methodologies, namely to apply an ontology of methodologies. In the discussed case, this process brought results in the first stage of constructing the taxonomy (dictionary) to define a semantic framework which then, being subject to the rules of knowledge engineering, took the form of an ontology.

During the construction of the framework the focus was placed on creating a framework of meanings for Scrum. The formal Scrum methodology is defined in the Scrum Guide. This document defines what is and what is not part of Scrum, and how Scrum combines these elements in the software development cycle. It deliberately does not specify how to implement Scrum in an organization. It can be concluded that the aforementioned document sets out a framework which, if implemented as described in the rules, allows the implementation of the Scrum model in any organization.

Table 1. Examples of the translation of the Scrum Guide into the controlled language

<table>
<thead>
<tr>
<th>A fragment of the Scrum Guide</th>
<th>Sentence representation in the controlled language</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Development Team consists of professionals who do the work of delivering a potentially releasable Increment of “Done” product at the end of each Sprint.</td>
<td>Every development-team-member is a professional that delivers a potentially-releasable-increment-of-done-product. Every potentially-releasable-increment-of-done product is an increment-of-done-product. Every sprint delivers-at-the-end a potentially-releasable-increment-of-done-product.</td>
</tr>
<tr>
<td>Only members of the Development Team create the Increment.</td>
<td>Every increment-of-done-product is created by nothing-but development-team-members.</td>
</tr>
<tr>
<td>Optimal Development Team size is small enough to remain nimble and large enough to complete significant work within a Sprint.</td>
<td>Every development-team that consists-of at-least three development-team-members and consists-of at-most nine development-team-members is an optimal-development-team.</td>
</tr>
<tr>
<td>Fewer than three Development Team members decrease interaction and results in smaller productivity gains.</td>
<td></td>
</tr>
<tr>
<td>Having more than nine members requires too much coordination.</td>
<td></td>
</tr>
</tbody>
</table>

It was concluded that the determination of how to implement Scrum is the domain of "best practices." Best practices have been developed and are made available in the form of training for Scrum Masters - individuals
responsible for the introduction of Scrum in organizations. It was proposed that the Scrum Guide should be treated as a framework and "best practices" as patterns operating within the aforementioned framework. These patterns, when equipped with appropriate semiotic layers, form a language of patterns, which allows for saving a specific implementation of Scrum in an organization.

This idea implies the need to examine the Scrum language of patterns. One of the research questions which emerged during the work conducted by the authors was the question about the optimal configuration of the Scrum patterns in a given environment of employees in an organization. It is suggested that this test should be carried out with the use of the simulation method which has been proposed in this article.

On the basis of the English version of the Scrum Guide and using the Fluent-Editor tool, a framework representation was created in the form of an ontology in a controlled language (Table 1, Figure 1).

The controlled language has the form of sentences, however, it is equipped with formal semantics expressed in logic. In the first stage, there were efforts to "literally" translate subsequent sentences from the source file into the controlled language. This ensured the consistency of the source text, and the text in the controlled language.

The process of the translation of the source text into the controlled text requires knowledge engineering (Table 1). Despite the expressiveness of the controlled language, equivalent to OWL2+SWRL, the full meaning of the text was not possible to be expressed directly in the aforementioned language. The restriction was revealed in the "soft" expressions referring to terminology intuitively understood by a human being, but which cannot be unequivocally recorded in logic - those with ambiguous meaning.

Fig. 3. Translation of the Scrum-GUIDE into the controlled language with the use of the FluentEditor™

It should be noted that due to the monotonicity of knowledge recorded in OWL2+SWRL, and due to the "spirit" of the Scrum Guide document, the rejection of ambiguous expressions does not lead to an error, but only to the loss of the so-called "soft" part of the knowledge. The rejected part requires a different method of representation - in this case, fuzzy logic was used (see section 4).

The resulting text in the controlled language, representing the structure of Scrum, was exported to an OWL/RDF file, by creating an ontology (Fig. 3).

While the structure of the Scrum model can be written in a formalism equivalent to OWL2+SWRL, the semantics of processes taking place during the operation of Scrum, in particular the patterns (best practices of using Scrum) as
well as the Scrum-SPRINT, require the application of a process description language. Describing processes goes beyond the scope of expressiveness of the OWL+SWRL format. The methods of knowledge representation, such as rules, the algebra of processes or the algorithms written in programming languages allow for the representation of knowledge about the patterns. The authors' work started with creating a library of rules stored directly in the form of a library of algorithms in Java. The library of patterns written in the Java language allows for the direct use of the Scrum ontology recorded in an OWL file. This process took place with the use of a reasoner, which provides appropriate functions for processing knowledge saved as a sum of the aforementioned ontology. Apart from the terminology (T-Box) discussed above, the second component of the sum is the configuration part (A-Box). A configuration example is shown in Figure 4.

| Pawel-1 is a development-team-member. |
| Pawel-2 is a development-team-member. |
| Pawel-3 is a development-team-member. |
| Pawel-4 is a development-team-member. |

 Anything either is Pawel-1, is Pawel-2, is Pawel-3 or is Pawel-4 or-something-else.

| Cezary is a Scrum-master. |
| Michal is a product-owner. |
| Darek is a product-owner. |

| Scrum-Team-110 is a Scrum-team that consists-of Michal and consists-of Darek and consists-of Cezary and consists-of Dev-Team-110. |
| Dev-Team-110 consists-of Pawel-1. |
| Dev-Team-110 consists-of Pawel-2. |
| Dev-Team-110 consists-of Pawel-3. |
| Dev-Team-110 consists-of Pawel-4. |
| Dev-Team-110 consists-of nothing-but either Pawel-1, Pawel-2, Pawel-3 or Pawel-4. |

Fig. 4. Example of a simulator configuration

4. The use of competence patterns for the construction of a fuzzy inference model for selecting project team members

The final stage of constructing the project framework was the use of competence patterns for the selection of project team members. At the core of the ongoing research lay the assumption that team cohesion (TC) will be evaluated from the perspective of three values, namely the complementarity of competence (CCIt), the degree of the typological match indicator to the role (TMIr) and the degree of the typological match indicator to the team (TMIt) of the team members, who have a strong mutual interaction (the meaning and the relevance of the "strong" interaction evaluation will be explained later in this work).

Complementarity of competence (CCIt) is understood as covering all the hard competences necessary, within the entire team, to carry out all the tasks defined in the project. The strength of interaction (SI) was defined and understood as a forced strength (determined by the team roles) of the bond between members of the team assigned to specific roles in the project team. This strength is determined by relational job position indicators (RWS), built for all the team roles based on a certain standard of running IT projects.

The degree of the typological match indicator to the role (TMIr) and the degree of the typological match indicator to the team (TMIt) are determined on the basis of a psychological tool, namely the Myers-Briggs Type Indicator (MBTI) which helps to determine the personality type of each individual team member, as well as the team as a whole. More information about MBTI can be found in 8. The degree of typological match is correlated with the accuracy of the typological role pattern (TRP) and the typological-relational role pattern (TRRP).
The article presents the structure of one of the indicators - the typological role pattern (TRP). The remaining structures are shown in [17]. The application of this indicator supports the process of selecting the personality type (according to MBTI), treating as the criterion the type of hard and soft competencies associated with a given role.

The construction of the TRP indicator involves determining (assigning) a preferred MBTI personality type to each project role \( r_i \). A group occupational psychology expert \( E_{P_k} = \{e_{p_{1}}, \ldots, e_{p_{l}}\} \), for \( k = 1, \ldots, l \), assigns for each personality type \( p_{t_i} \), for \( i = 1 \ldots 16 \), a score of 0 to 1 (in a range of decimal values) evaluating its adequacy for the project role. In this way, the matrix of the typological job position indicator \( A_{TRP} \) is formed, in the \( n \times m \) size, where \( n \) is the number of roles in a given team model, and \( m \) is the number of the typological indicator evaluated in the context of job position roles,

\[
A_{TRP} = \begin{bmatrix}
a_{11} & \ldots & a_{1n} \\
\vdots & \ddots & \vdots \\
a_{m1} & \ldots & a_{mn}
\end{bmatrix},
\]

where, for \( i = 1 \ldots n, j = 1 \ldots m \).

Then the standardization of the indicator of the job position relationship follows by averaging the results of expert evaluations,

\[
\frac{1}{l} \sum_{k=1}^{l} a_{ij}^k,
\]

for \( i = 1 \ldots n, j = 1 \ldots m, \ k = 1 \ldots l \).

For the construction of the project framework, the relational position pattern (RPP) and the typological-relational job position indicator (TRRP) were used. The first one determines the strength of interaction between roles in the project team. The second shows what type of personality should characterize people working together as a team, and whose job position relationship is characterized by high strength interaction.

In the case of Scrum as a management method, the construction of TRP requires the project role of the Team to be made more specific. While this role is assumed to be an individual unit, it contains a set of individuals, specialists, with assigned roles regarding business analysis, the development of IT system architecture, programming, testing, and the implementation of developed solutions. Therefore, the typological job position indicator for the Scrum model does not describe three roles (according to the Scrum Guide) but seven project roles, where the Team role includes five specialized roles associated with the performance of project tasks. In effect, the developed model assumes a set of project roles \( R = \{r_1, \ldots, r_7\} = \{\text{Product Owner, Scrum Master, Business Process Analyst, Designer, Programmer, Tester, Implementer}\} \), where the roles \( r_3-r_7 \) are sub-roles of the Team role. A set of sixteen personality types (according to MBTI) \( PT = \{p_{t_1}, p_{t_2}, \ldots, p_{t_{16}}\} \) was also adopted.

For the adopted assumptions, example of the TRP indicator is presented below:

\[
A_{TRP} = \begin{bmatrix}
0.3 & 0.2 & 0.3 & 0.4 & 0.9 & 0.8 & 0.5 & 0.6 & 0.6 & 0.3 & 0.6 & 0.9 & 0.8 & 0.5 & 0.7 & 0.8 \\
0.3 & 0.2 & 0.2 & 0.4 & 0.4 & 0.2 & 0.2 & 0.2 & 0.7 & 0.9 & 0.8 & 0.8 & 0.2 & 0.2 & 0.2 & 0.2 \\
0.8 & 0.3 & 0.3 & 0.5 & 0.4 & 0.3 & 0.2 & 0.3 & 0.9 & 0.5 & 0.5 & 0.8 & 0.7 & 0.5 & 0.4 & 0.5 \\
0.9 & 0.3 & 0.2 & 0.5 & 0.6 & 0.3 & 0.2 & 0.2 & 0.8 & 0.2 & 0.1 & 0.4 & 0.5 & 0.2 & 0.1 & 0.1 \\
0.6 & 0.5 & 0.6 & 0.7 & 0.4 & 0.2 & 0.2 & 0.5 & 0.4 & 0.8 & 0.8 & 0.3 & 0.3 & 0.2 & 0.2 & 0.2 \\
0.5 & 0.7 & 0.9 & 0.5 & 0.2 & 0.7 & 0.8 & 0.8 & 0.5 & 0.7 & 0.8 & 0.5 & 0.2 & 0.5 & 0.3 & 0.3 \\
0.5 & 0.6 & 0.7 & 0.3 & 0.2 & 0.3 & 0.4 & 0.2 & 0.4 & 0.7 & 0.8 & 0.4 & 0.2 & 0.2 & 0.2 & 0.1
\end{bmatrix},
\]

The obtained values, representing expert knowledge and their assessment of the quality of the fit for the personality type to the role, specifying the TMir are grouped into four linguistic sets: bad, average, good, very good, assuming the interval variability to be: for bad \([0 - 0.3]\), medium \([0.2 - 0.5]\) good \([0.4 - 0.8]\), very good \([0.7 - 1]\).

The RPP indicator allows the project roles to be represented by a matrix expressed below (4):
The elements of this matrix determine the strength of the interaction SI between roles in the team and are grouped into three linguistic sets: weak, medium, strong, assuming the interval variability to be: for weak \([0 – 0.4]\), medium \([0.3 – 0.7]\), strong \([0.6 – 1]\). Taking into account that in the case of Scrum teams, the strength of the relationship between all roles is high, it was decided to implement the TRRP indicator for all combinations of roles (21 patterns). An example interpretation of one TRRP indicator for the roles of Scrum Master and Product Owner is presented below (5).

The elements of this matrix determine the TMIt degree of typological matching to the team (co-workers) and are grouped into four linguistic sets: weak, average, good, very good, assuming the interval variability to be: for weak \([0 – 0.3]\), medium \([0.2 – 0.5]\) good \([0.4 – 0.8]\), very good \([0.7 – 1]\). The need for these indicators results from the necessity to support value judgments and to rank the evaluation of team members.

Variables TMIr, TMIt and CCIt values, beyond being the basis for the designation of team’s cohesion, can be also implemented for team member choice optimization from a given set of candidates. For this purpose optimization model can be based on decision making in the so-called fuzzy environment, consisting of fuzzy objective, fuzzy constraint, and fuzzy decision.

**5. Conclusion**

The article presents the issues of constructing simulation systems of project team work (SPSM). They are systems relevant to the development and planning of the work of these teams (e.g. IT support organization). The construction process of such a system has been presented, pointing to the importance of its high-level architecture for the identification of variables. These variables are the basis for determining the functionalities of the system, and at the same time, they create the conditions for defining and implementing system processes for the needs of high-level architecture.
The authors have encountered difficulties in defining and implementing these processes. Therefore, they proposed the use of an ontological-fuzzy project framework. The selection of the ontological framework resulted from the need to combine the knowledge on how the environment functions with the knowledge of the processes of its construction. Whereas, the selection of the fuzzy framework resulted from the need for a multivariate evaluation of staff competence. The project framework proposed to support the construction of SPSM was verified with the use of the Scrum management method (its processes and roles). The Scrum processes combined with the environment construction processes allowed for the evaluation of the consistency of Scrum and SPSM processes. The Scrum roles allowed for the verification of the applicability of the competence indicators for the evaluation of two Scrum roles: Scrum Master and Product Owner.

Despite the application of the project framework, it seems that the further development of SPSM requires this framework to be made more precise. The project framework facilitates the SPSM design process but it is difficult to accurately evaluate its applicability before the entire SPSM design process is closed. While the use of ontology shows the relationship between the SPSM and Scrum and creates conditions for the implementation of Scrum, the evaluation of the applicability of the competence indicators is much more difficult. It appears that full utility of these indicators will be possible after the completion of the process of the fuzzy selection of members for the project team. Data provided by an IT company showed the state of Scrum (its sprints). However, no data was provided regarding the selection of team members, and hence the applied competence indicators support the evaluation of the current state of the project rather than the selection process of its composition.

Therefore, the authors plan to focus further work on the development of the project framework presented in this work. It seems that for the process of selection/application of this framework, a different method of project management will have to be modeled. The use of RUP13 (Rational Unified Processes) is considered as one possibility. However, the key factor for the selection of a method for verifying the project framework will be a complete compendium of knowledge not only about the selection of team members, but also on monitoring their behavior during the project. Unfortunately, the insufficient scope of such projects and access to knowledge about their course are serious limitations to this approach.

References

