INTEGRATION OF CLASSICAL AND AGILE PROJECT MANAGMENT METHODOLOGIES BASED ON ONTOLOGICAL MODELS

Jan WEREWKA, Piotr SZWED, Grzegorz ROGUS

AGH University of Mining and Metallurgy, Department of Electrical Engineering, Automation, Information Technology and Electronics.
Automation Faculty, Laboratory of Information Technology.

Summary. In the paper the problems of IT project management and trends of project management methodologies development are discussed. The two base categories of project management methodologies are presented: classical based on project plan and agile based on product streaming to increase its value at the end of release iteration. As stated in the work, it is advisable to use both methodologies or to integrate them within an IT enterprise. The article proposes an approach to the problem of integration of project management methodologies consisting in building their formal ontological models and performing integration of these models based on techniques of concept mapping and ontologies alignment. Ontologies of the PMBOK and the Scrum, the two most popular project management methodologies representing classical and agile approach, are developed. The results of their integration will constitute a coherent formal description defining mapping of elements of both methodologies. They may be useful for IT enterprises running projects in heterogeneous environment and can be a basis for future integration of both methodologies.

Keywords: IT projects management, Scrum, PMBOK, ontology, modeling.

1. INTRODUCTION

The development of Information Technology sector, which is highly significant from the economic and strategic point of view, depends decidedly on a successful management of various IT projects in this industry. The classical (traditional) methodologies fulfill a recognized role in this process. However, we observe lately a vivid development of the management methodologies known as agile.

The question is whether both classes of methodologies can be successfully used in an IT company and is it advisable to integrate them. This paper focuses on companies for which it may appropriate to use both methodologies or to integrate them. The assumptions are as follows:

- The suggested solutions apply to IT enterprises whose business activity is mainly based on projects realization.
- The IT companies provide an environment for conducting many projects at the same time.
- At the lower management levels the projects can be conducted, using various management styles and production techniques.

The last point is significant, for it allows managing running simultaneously projects using traditional or agile methodologies. The necessity of applying both types of methodologies stems from the following issues:

- IT companies report an increased productivity and improvement of business profitability indicators in case of agile methodologies.
- The classical management methodologies, based on planning, also grow popular. This results from their universal character, a company's need for switching from operational to project activity and the requirement of having a common reference to other methodologies.
- Only some projects conducted according to the agile approach in IT enterprises can bring the expected results.
- Some IT projects must be plan-oriented due to various reasons, such as external limitations.

In many studies the traditional methodologies are contrasted with the agile ones. On one hand, it is important to demonstrate their differences, and on the other proving that it is impossible to integrate them seems wrong.

The question arises whether it is expedient to integrate them. In case of a group realizing a project according to one methodology it does not seem profitable. Yet, from the standpoint of a company carrying out diversified projects, such integration may appear perfectly justified.

It appears that discussions on integrating project management methodologies lack of precise models that could be a base for indicating common elements and differences. The authors suggest that ontological models, defining formal concepts and relations in the analyzed methodologies, could serve this purpose.

In the article it is proposed to integrate classical and agile project management methodologies, based on integration of ontological models representing them. The presented solution consists of three phases: to construct ontological models of the selected methodologies, to suggest a mapping between them and to integrate ontologies.

The paper is organized as follows: chapters 2-6 present the specificity of the IT industry and typical problems of projects management in this business. They also demonstrate differences between traditional and agile methodologies as well as attempt to prove the necessity of standards integration. Chapters 7 and 8 present tools: ontological models and techniques of integrating them. In chapters 9 and 10 there are described ontologies of the most popular traditional methodology – PMBOK, and the agile one – Scrum. Chapter 11 contains summary and conclusions.

2. THE SIGNIFICANCE AND SPECIFICITY OF THE IT SECTOR

The IT industry is related to information technologies, which are rapidly evolving, and there are more and more new application areas. IT industry has a high market value and enormous strategic importance, due to the fact that every sector of business is dependent on information technology.

The issue of project management in IT companies is also of strategic importance. The development of information technology industry in Poland is at about 12-14%.

According to data presented in [1]:

- In 2007 the Polish IT industry employed 295,000 people.
- The market of Polish IT business is highly fragmented; there are over 5,000 active companies.
- According to IDC data in 2007-2011 in Polish IT sector there will appear about 109,000 new workplaces and 1,700 new IT companies.

Although, the market of Polish IT business is very fragmented, in 2006 we observed a considerable market consolidation due to fusions of companies.

Polish IT industry is growing very dynamically [2]. According to IDC data in 2007-2011 in Polish IT sector there will appear around 109,000 new workplaces and 1,700 new IT enterprises. In the next four years the Polish IT market will generate tax revenues of 11.2 billion PLN to the budget and will increase Polish GDP by 31.9 billion PLN.

In 2007 IT investments in Poland reached 25.3 billion PLN. IDC predicts that in years 2006-2011 the growth of yearly expenditure on Polish IT sector will arrive at 12.8%, 9.8% of which will constitute the expenditure on software. Comparing to other countries, whose investments on IT get to 2.5% GDP, in Poland they reach 2.2% GDP.

Expenditure on software makes up 15% of all IT expenditure in Poland, and 38% of employees in this sector create, distribute, deploy and maintain software.

Due to the rapid development of information technologies and a vast diversity of their application, project management methods are changing as well. In Poland there is a great need for effective project administration and management. This is caused by challenges, which the IT industry faces and changes in customer attitudes towards the purchase of software. Efficient project management will contribute to further development of IT companies. Otherwise, along with wages growth, Polish companies will be of little interest to investors.

The situation on the market of software producers is altering as well, as customers understand that buying software not always brings measurable profits [3]. More and more often it is required that the software is error free, interoperates easily with other programs and is provided as a service via the Internet.

3. IT PROJECTS PROBLEMS

With so much economic and strategic importance of the IT industry, it is indispensable that the software development process is effective and assures a certain quality. However, there often appear major problems in IT projects management, and they are not resolved successfully. It is testified by data collected by the Standish Group - an American consulting company that prepares one of the most recognized statistic reports concerning IT projects success or failure, known as the Chaos Reports. They illustrate IT projects realization in the USA. The statistics focus on a project success based on a popular triangle of constraints, according to which the project is successful if it was realized:

- on time (schedule),
- at the planned budget (costs),
- with the required features (scope).

In the Standish Group report [4] there are enumerated three possible types of project termination:

- The project is successful if it was completed on time, its costs did not exceed the planned budget, and it realized all requirements specified at the beginning.
- The project is partly successful if the product developed in the course of the project was completed and accepted, but at least one of the project's conditions was not met – deadline, budget, scope or quality.
- The project is unsuccessful if it was not completed or its final products were not used.

The research of the last 15 years prove that a great number of projects turn out to be unsuccessful and in the last few years the situation has become even worse (Fig. 1).

The report indicates also that lately the projects results have deteriorated, as compared to year 2006. It is a weird phenomenon, as the number of projects management experts has grown and there are better tools and trainings offered. Additionally, the information of management methodologies is more complete and the cooperation rules among project stakeholders improved. Alternatively, due to the dynamic development of the IT sector, the tools and solutions applied are often new are not always fully recognized. What is more, the systems are more complex and the products delivery time is often shortened.

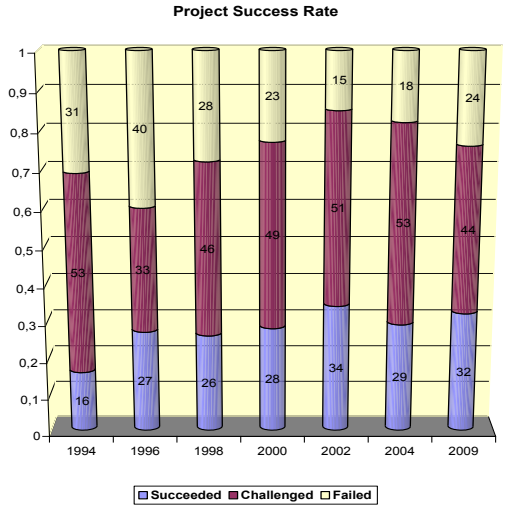

Fig.1. IT projects success rate, according to the research of Standish Group Inc.

The principles of working out the statistic data by Standish Group are often questionable [5]. The doubts stem from the way of defining the project success; in fact, it is not always determined by the three factors given by the researchers. The project may be successful, due to other aspects, such as customer's satisfaction, usability, profit, quality of the products delivered, or risk level.

Despite all uncertainties about the research and statistic data, they still are a point of reference for the evaluation of projects outcome in the IT business. Based on their results, it is clear that IT projects are the type of ventures of a very high risk and failure level.

Standish Group reports contain also respondents' information, defining the conditions of a success. Tab. 1 presents first ten of the conditions for years 1995, 2001, 2006.

The most common reasons for project failures are: incomplete requirements, lack of client's involvement, requirements changes, lack of managerial support, lack of competencies in the given field, insufficient human resources, client's unrealistic expectations, unclear goals, unfeasible deadlines and new technologies.

Tab. 1

First ten conditions of an IT project success, according to Standish Group.

1995	2001	2006
Client's involvement	Managerial support	Client's involvement
Managerial support	Client's involvement	Managerial support
Clearly defined requirements	Experienced project manager	Clear business goals

Proper planning	Clear business goals	Scope optimization
Realistic expectations	Minimized scope	Agile process
Shorter intervals between milestones	Standard programming structure	Experienced project manager
Employees' competencies	Clear basic requirements	Budget management
Responsibility	Formal project methodology	Competent human resources
Clearly defined goals and demands	Realistic estimation	Formal project methodology

Source: Standish Group [4].

The question is – what are the basic difficulties in IT projects realization, as compared to other sectors? The most evident are:

- The software is difficult to present and it is not easy to explain its requirements.
- It hardly happens that the same software is created more than once – usually there is no analogy to similar, already existing systems.
- Conducting an IT project is a complex and risky undertaking, due to e.g. technological changes during the project course.
- While the software is being created, there often is a need for changes, originating from imprecisely defined requirements.

In order to deal with the above mentioned problems, considering the IT sector specificity, it is necessary to: work out and apply proper project management methods, employ proper methods of software production, allowing frequent quality verifications, as well as improve managerial and production processes adjusted to organizational needs and capabilities.

4. AGILE METHODOLOGIES AS A REMEDY FOR THE PROBLEMS

The demonstrated problems associated to IT projects realization triggered a quest for solutions more adjusted to the sector's needs. If a client ordering software is unable to define his requirements clearly, maybe it would be better to give up the classical methodologies, based on a precise project plan. Preparing and maintaining a detailed plan is costly. In return, a client should be allowed to make changes in a standardized way by adapting the project process, so that all his needs are taken into account in the final product.

In the adaptation (agile) approach planning is based on the already existing product features, trying to grow at every stage its value. Agile methodologies accept and expect changes (product verification is carried out because of the vague requirements). It is assumed that project teams are able to realize all system features, and thus they should have

interdisciplinary knowledge, needed to complete the project. Moreover, the teams should be self-organizing.

The question arises, how the agile, change-oriented approach to project realization can be profitable, while changes are believed to be very costly. In some types of IT projects this situation is not likely to occur. Tab. 2 presents rough costs of changes in case of a construction and IT venture, to demonstrate the differences.

Tab.2.

Main costs components of changes in construction and IT ventures.

Project costs	**Construction project**	**IT project**
Planning and designing	Medium or high	High
Human resources	Medium	High
Tools	Medium	Medium or high
Materials	High	Low or none
Demolition of a whole construction or its part	High	Low or none
Demolition materials utilization	High	Low or none
Architecture change	High	Medium

Source: own.

The major features of IT projects, as compared to typical, e.g. construction projects, are: difficulties in specifying client's requirements (changing requirements), or fairly low cost of changes. Complexity and innovation is usually higher in IT projects.

Nevertheless, it is not always possible to apply agile methodologies in IT projects. In literature there are many studies defining criteria of methodology selection. One of the most recognized is the radar chart drawn up by Barry Bohem and Richard Turner [6], featuring five crucial criteria of choosing traditional or agile methodology:

- Dynamics of requirements changes. The dynamics illustrates percentage of requirements that can alter within a month. If the percentage of changes is high, e.g. 50%, then the agile methodologies are suggested. Otherwise, the classical approach may be sufficient.
- Team. This axis presents a proportion of team members having basic, medium or expert skills in the given area. In agile methodologies it is advisable that not many people of the beginner level work on the project. Conversely, in the classical approach, in which a detailed documentation is provided, more inexperienced team members can take part in the project. Additionally, other studies on agile methodologies presume that the team is interdisciplinary – more people share the same knowledge - in order to produce software whose quality (properties) can be easily assessed by the client. Therefore, a team of experts with disjoint knowledge is not recommended for agile methodologies.

- System criticality. Some software needs to meet extremely rigorous safety or quality requirements, because its potential failure or malfunction could cause immense loss. In case of very high safety and quality requirements it is more advisable to apply the traditional methodologies.
- Team size. The agile approach assumes that project teams are rather small, considering their way of working and communicating with one another.
- Organizational culture. In companies where the organizational culture is not normalized (chaotic) it is easier to introduce smaller teams working according to agile methods. Yet, enterprises with an efficient project organization may not benefit much from switching to the agile approach.

The presented division is rough and does not include a few additional, but crucial factors:

- Contract type. In agile approach it is believed that trust and cooperation is more valuable than contract negotiations. Therefore these methodologies base on agreements of a low risk for the seller, such as contracts with reimbursed costs (where the seller is paid according to actual costs), or time and material contracts. The higher the seller's risk is, the more probable will be choosing traditional methodologies, especially in case of contracts with fixed price or single payout, being an agreed total of the product delivered.
- Schedule type. If a project schedule assumes fixed tasks deadlines, resulting from the need of coordinating work between different project team members, then it is more convenient to apply a traditional approach.
- Budget type. A budget arranged unevenly in time, forcing varied intensity of work, favors traditional methodologies, while a budget financing the project evenly and regularly favors the agile approach.
- Quality level. Meeting the requirements associated with standards, licenses or certificates may compel using the classical methodology.
- Approach to project risk. Agile approach assumes a rather low project risk. If the risk is high, the traditional methods should be applied, because they require developing mitigation plans, avoiding risks or defining emergency procedures.
- Organizational structure realizing the project. Organizations using highly specialized units will prefer more traditional methodologies of project management.

The above mentioned arguments prove that it is necessary for classical and agile methodologies to coexist or it may be useful to combine them into one methodology. Thus, it is necessary to model the two methodologies, to define their similarities, differences and ways of their possible integration.

5. TRADITIONAL AND AGILE METHODOLOGIES OF IT PROJECT MANAGEMENT

Methodologies of IT projects management belong to a area of knowledge that is developing dynamically. In practice there are many solutions based on the traditional or agile approach to project management. There are also many research concerning the use of both methodologies in an IT company. There are many popular classical methodologies as: PMBOK, PRINCE2, COBIT, etc., however, PMBOK will be used as a reference, as it is most widespread. It is an open PMI project management methodology, and an ANSI standard as well.

On the other hand, Scrum is proved to be the most popular agile methodology; according to VersionOne Inc. 2008 research [8], it scored 49% of the market share among 3,000 respondents (on the second place there was the hybrid Scrum-XP, 22.3%), while other methodologies, such as AgileUP, FDD, Lean Development, DSDM, OpenUP, Agile Modeling and Crystal gained together about 10% of the market share.

The main difference between traditional and agile methodologies of management is a different approach to planning and developing a product within a project.

Traditional methods are plan-oriented. There are specific requirements defined, and on their basis a schedule is determined and the project cost is estimated. The project's goal is to deliver a product, meeting all defined requirements. Planning is the crucial part in this type of approach; tasks are transferred from the planning processes to the execution processes (realization as planned). Project progress is verified as compared to the plan, in the case of divergence, some corrections are planned and applied.

Agile methodologies base on the value of the delivered products, while cost and time interval is predetermined. After the declared time, the team is supposed to deliver a product of the highest possible business value. In this approach there is a list of requirements, determining changes to the existing software prototype. Project manager (or a similar project role) maximizes the product value by proper selection of features for the current iteration, and eliminates possible obstacles that the development team can face. This allows early verification (validation) of requirements.

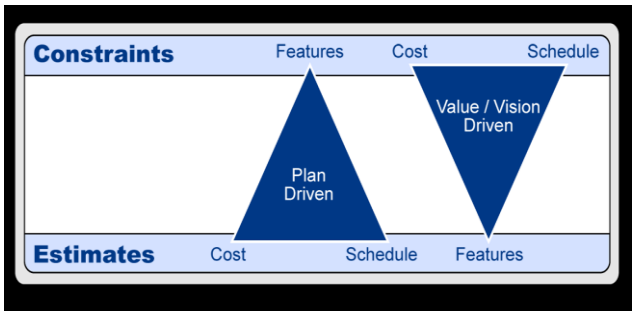

Fig.2. Comparison of the main constraints and planning results in traditional and agile methodologies.

In traditional methodologies the requirements that the product should meet are the starting point. They constitute the grounds for project budget and schedule.

In case of agile methodologies, the goal is to enhance the product value, as compared to its current state, at the defined costs (resources) and within the agreed time limit (Fig.2).

6. NECESSITY OF THE STANDARDS INTEGRATION

Following observations and practice in projects management in IT companies, as well as literature examples, one can notice that companies often drift in the area of project management and do not achieve the expected results. Authors experience indicate that for IT companies the best way is to concentrate on a certain framework joining traditional and agile methodologies. After scaling and configuring, the framework elements should be adjusted to the management of a specific project [9]. Moreover, on the level of an enterprise it is advisable to concentrate only on the most popular management methodologies and use them even for reasons of organizational support and dedicated tools used in the enterprise. The suggested approach should also allow a unified view from higher levels of management on the projects realized according to different methodologies. Such perspective will make it possible to compare, assess them and to make decisions more easily. This approach is also justified by observations, proving that in most companies the applied project management methodologies are not entirely consistent with their original specification. It is important that the suggested solutions are adapted to the size and complexity of projects, and to various limitations e.g. originating from company organizational maturity.

The issue of integrating projects management is of great importance for IT enterprises from the economic and logistic point of view. Economic and scientific literature describes a noteworthy example – the fusion of two IT giants, HP and Compaq. In this case a suitable solution for projects management was sought as well [10]. This example proves that it is not always possible to select a suitable methodology for every project, as it is often believed, because of enterprises' actual needs and conditions.

To sum up, it may appear practical to integrate and scale methodologies of project management and develop tools that would facilitate the process of selecting and deploying those methodologies. The following needs and situations may prove it:

- Realization of various types of projects, whose efficiency depends on the enterprise's organizational structure.
- For a company's executive board it is necessary to have an overall view of the realized projects, and the tools and solutions applied.
- For a company's staff it is necessary to have some standards defining procedures, roles, responsibility scope, etc.
- IT enterprises are willing to introduce the agile methodologies (and they apply their own adaptations). They tend to seek target solutions and reach them without affecting the productivity.
- Absence of unified definition of resources and processes of project management in companies. Such companies search for tools allowing flexibility of the definitions

and easy adaptation to the specific needs. There is a need of such tools on the market.

- Fusions of companies having different standards of projects management (the example of HP and Compaq).
- Realization of projects in heterogenic, virtual structures of organization (e.g. international or EU projects).

In enterprises of the IT sector, whose main activity is project-based, it is believed that employing the verified management methodologies as PMBOK [7] or Scrum [11] will improve software production process and its competitiveness.

There are some obvious differences between the classical and agile methodologies. The traditional approach focuses on classifying and identifying project processes, which is consistent with the holistic approach proposed in Enterprise Architecture Framework [12]. On the other hand, the agile methodologies do not define processes explicitly, but determine strict principles, procedures (that could be treated as processes) and rules of project organization.

The issue of selecting an appropriate methodology and defining project organization adjusted to local conditions is very complex. It is necessary to consider limitations and circumstances, resulting from the company's organizational structure, available resources and knowledge. Then again, it is expected that the defined architecture of project management processes should be optimal or suboptimal, regarding such factors as value added or project risk.

In the process of introducing, adapting and scaling specific project methodologies there is a gap; there are no tools allowing to assess early enough whether the chosen approach and project process architecture is appropriate for the enterprise and the project type. A wrong choice in this field can result in a project failure or exceeding the initial limitations, as schedule or budget.

An answer to the question whether the organization of project management processes is proper appears after a project finishes, and becomes a part of the company's historical knowledge (lessons learned). Using the information on the best solutions and the history of successes and failures is beneficial, since it allows decreasing project risks and better estimating costs and the realization time in the future. However, it may also appear disadvantageous, if it leads to a lack of openness to changes and innovations, for fear that the already achieved productivity might be lost.

7. ORGANIZING PROJECT MANAGEMENT KNOWLEDGE WITH ONTOLOGIES

Project management methodologies are most often specified in a descriptive form. Such form of presentation seems insufficient, because there may appear incoherencies, ambiguities and difficult to catch differences of meanings. To avoid such drawbacks, in this work the usage of ontologies for formal description of project methodologies is proposed.

There exists many definitions of the term ontology. A popular one, is the definition given by Gruber [13]: “An ontology is a specification of a conceptualization, That is, a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents.”

Ontology constitutes an abstract a model of entities existing in a selected domain obtained by identifying the essential concepts associated with it. This means that for some domain there have been identified existing concepts (objects, events, conditions, etc.) and relations between them. The term *formal* means that the model must be machine readable, which excludes natural language. The term *specification* means that the definitions of the concepts and relationships must be clearly stated, the term *common* refers to the fact that the knowledge contained in ontology is to be acceptable by all users. Gruber definition in its simplicity is trying to include most aspects of ontologies.

Ontology may be also understood, as a model represented by set of concepts within a domain and the relationships between those concepts [14]. The basic components of the model are:

- Individuals (instances or objects).
- Classes (sets of individuals, concepts, types).
- Attributes
- Relations.
- Functional terms: complex structures constructed with the selected relations that can be used in place of an individual term in a statement.
- Axioms, in that case assertions determining features of selected classes, which enable to distinguish them.

Considering only the practical perspective of modeling of a certain domain, the ontological models can be perceived to be similar to object-oriented models, as UML [15], and especially to class structure diagrams. They, however, offer such advantages as a rich set of tools and advanced techniques supporting modular model building, model exploration, reasoning, and also mapping and integration of models. Examples of tools are ontology editors, e.g.: Protégé [16], reasoners, as Pellet and Fact+, software libraries supporting manipulation of ontological models, e.g., Jena library [17].

8. PRINCIPLES OF THE ONTOLOGY BASED INTEGRATION OF METHODOLOGIES

In the article we propose to integrate project management methodologies using ontology matching and integration techniques. The described approach consists in three steps:

- building ontological models of the selected methodologies,
- elaborating mappings between ontologies, what can be done manually or with use of automatic ontology matching tools,
- ontology integration.

The description of PMBOK and Scrum ontologies is given in sections 9 and 10. In this section techniques of ontology matching and integration is discussed.

8.1. ONTOLOGY MATCHING

Ontology matching is the process of determining correspondences between concepts belonging to two ontologies O and O'. The term ontology matching or ontology alignment also refers to specification of such correspondences.

The formal definition of ontology alignment [18] defines it as a set of tuples: (e, e', R, n), where

- e and e' are entities (classes, properties, individuals) belonging to ontologies O and O'.
- R is the relation between e and e', e.g.: equivalence (=), subsumption ($\sqsubseteq$), disjointness ($\perp$) and overlapping ($\bullet$).
- n is the confidence factor, typically in range [0,1].

The book [18] discusses several techniques that can be used to determine alignment of ontologies. They include name based, structure based, extensional and semantic techniques.

Name-based techniques establish an alignment by comparing strings: entity names (or URIs) but also labels and comments. They can use several metrics to calculate string distances between entity names (for example the required number of deletions and insertions of characters that transform one name into another). For distances above a certain threshold an appropriate tuple with the confidence factor equal to the calculated and normalized distance is added to the alignment.

Structure based methods determine the correspondence of entities by analyzing their internal structure (relations and data types used to express properties) as well as their position in the ontological hierarchy.

Extensional techniques can be applied in a certain situation: if individuals are available. Such situation may occur, for example, if concepts in analyzed ontologies can be defined by indicating sets of records in data bases. This gives an opportunity to match ontological classes that are treated as intentional specification of individuals based on relations between sets of individuals: equality of sets corresponds to equivalence, inclusion of sets to subsumption of concepts, not empty product of sets to overlapping and the empty product to disjointness.

Semantic-based techniques consist in determining the correspondence of entities of two ontologies by comparing their meaning with respect to an external formal specification that is usually represented by an upper ontology. Those techniques require an initial step where compared ontologies are *anchored* in an external ontology, i.e. their entities are assigned with a concepts appearing in an external ontology. This can be performed manually or automatically with use of described earlier techniques.

8.2. INTEGRATION

The term ontology integration has not precise meaning. In the work [19] there were identified three different situations where the term integration is used:

- Integration consisting in building a new ontology with reusing existing ontologies by extension, specialization, adaptation and concept mapping.
- Integration of ontologies by merging different ontologies about the same subject into a single one that "unifies" all of them. Very often, an introduction to ontology merging is establishing of a mapping between concepts. Identified relations became then axioms of the resulting ontology.
- Integration of a number of ontologies inside an application using them not accompanied by changes in ontologies.

Sowa [20] discusses the problem of ontology integration in the context of using them information systems. Integration is defined as the process of finding commonalities between two different ontologies A and B and deriving a new ontology C that facilitates interoperability between computer systems that are based on the A and B ontologies. The new ontology C may replace A or B, or it may be used only as an intermediary between a system based on A and a system based on B.

Three levels of integration are distinguished: alignment, partial compatibility and integration. Alignment is treated as the weakest form of integration, supporting mainly classification and information retrieval. Partial compatibility can require changes in A and B ontologies to enable better cooperation of the systems. And finally, unification means full compatibility of both ontologies and full cooperation of both systems (for each operation in the first system, there exists a full equivalent in second one).

Agile and classical project management methodologies are developed by distinct organizations. If assumed that the methodologies are specified by separate ontological models, it would be difficult reach the unification level, because this might require changes in the methodologies itself. In our works two research goals are defined: ·

- For short term horizon, development of mapping proposals between methodologies models and formalization of ontology describing the mapping; this corresponds to the alignment level.
- In a long term horizon, developing changes in the methodologies to reach a partial compatibility level and presenting proposals to PMI community for verification.

Definition of mapping between methodologies will be done manually and also with use of structural and semantic ontology matching techniques.

8.3. LAYERS OF THE MODEL

The developed and being under construction ontologies are grouped in three layers (Fig. 3. The lowest one (predefined ontologies) contain base ontologies and models of basic concepts of the problem domain as: artifacts, roles, processes, tools, techniques and events.

The next layer defines structures of classes (aggregates). The structures contain large PMBOK, Scrum ontologies (both including several hundred classes linked by relations) and the models of organization structures.

The highest integration layer encompasses: mapping of roles, events (modeling here project phases, stages or planning horizons), changes and processes. Due their complex nature, they are defined also as ontologies. This layer also contains ontological scalable process descriptions, which constitute coherent sets of processes, formed from the initial model by removing some selected processes and artifacts (input and output flows).

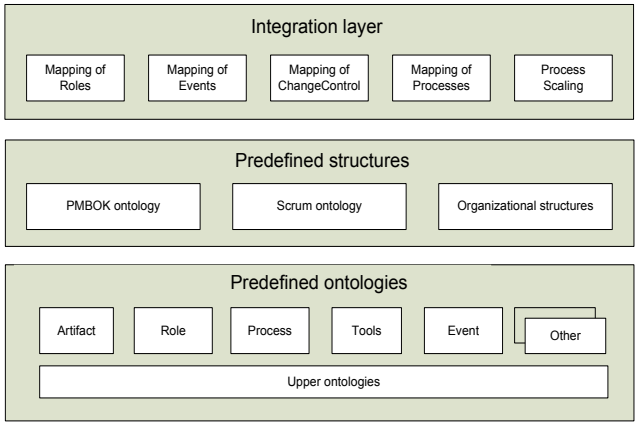

Fig. 3. Layers of ontological models for integration of project management methodologies

9. ARCHITECTURE OF THE PMBOK ONTOLOGY

The PMBOK combines the knowledge of proven traditional and widely applied practices of project management. The goal of the PMBOK ontology is to provide a formal description of knowledge about the process of project management based on the methodology. Ontology construction has been limited to issues of managing a single project. In the paper we discuss only selected issues related to the constructed ontology and focus on the presentation of main classes forming the skeleton of an ontology.

The PMBOK ontology contains five main groups of classes: *Artifacts* (objects that are produced and consumed by project management processes), *Roles* (classifying persons or groups of persons according to their functions), *ProjectFramework* (contains all classes directly assigned to project), *Processes* (specified management processes) and *Techniques&Tools* (all tools and techniques that can be applied during processes execution). This taxonomy is supplemented by a subsidiary class: *Properties* (being a base for classes without defined structure). In *ProjectFramework* there are two main classes: *ProcessGroups* and *KnowlegeAreas*. They represent two different classifications of processes: the first related to their assignment to various stages in the project lifecycle, the second based on their domain.

9.1 EVENTS

Ontological classifications distinguish concepts describing objects that are durable and time independent (*Continuant*) and objects that are characterized by temporal relations (*Occurant* or *Event*) [21]. In the upper ontology of events used in both models PMBOK and Scrum, there are distinguished events having a certain duration (*DurableEvent*), events occurring at a certain moment, but having no duration (*InstantenousEvent*) and events repeating with a certain frequency (*PeriodicEvent*). For those events specific relations are defined describing their environment (*hasAgent*, *hasObject*, *hasLocation*), temporal properties (*hasDuration, hasFrequency, hasOccurenceTime*) and temporal relations (*after, before, during, finishes, overlaps, starts*) as proposed in [22].

Those relations are used for precise definition of events: their participants, durations and mutual temporal relations. In the PMBOK ontology events are used for modeling the project life cycle, partition of the project into phases and classification of their stages.

One of the main concepts in the PMBOK methodology is the concept of project represented by the *Project* class. Project is "a temporary endeavor undertaken to create a unique product or service" [7]. Each project has a definite beginning and end. End of the project is reached when project's objectives have been achieved or when the project is terminated because its objectives can not be met.

Each project has limited physical resources *Resource*. The main resources of project are: *People*, *Equipment* and *Material*. Other resources are modeled by *AnotherResource* class and represent know-how, organization knowledge, headquarters, etc.

Each project has four basic parameters: scope (*ProjectScope*), cost (*ProjectCost*), time (*ProjectDuration*) and objectives (*ProjectObjective*).

Project's life cycle in the PMBOK contains one or several phases. The PMBOK phase can be considered as a full project (a project consists of one project phase) or subproject (a project contains more project phases). The situation is modeled by relation *isComposedBy min 1 Phase*. In the PMBOK methodology three phase-to-phase relationships are distinguished:

- sequential (next phase starts after the previous one is finished)
- overlapping (next phase starts during another phase)
- iterative (planning for the next phase is performed while the current phase is executed)

Each phase contains 5 events: *PhaseInitiating, PhasePlanning, PhaseExecuting, PhaseControling&Monitoring,* and *PhaseClosing*. These events define time periods during which different interacting activities occur. The activities belong to the following process groups: initiating, planning, executing, monitoring&control and closing. Figure 4 illustrates relations between the events.

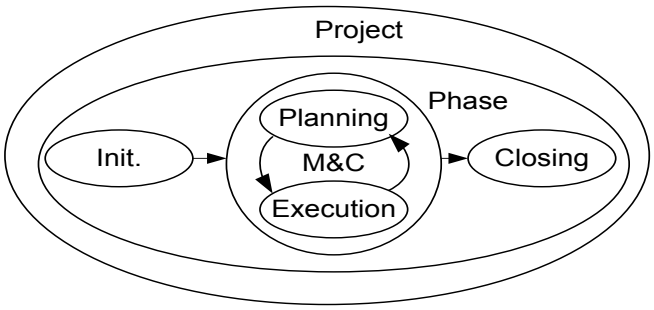

Fig. 4. Events in the PMBOK methodology

PhaseInitiating is an event which starts each project phase and is executed only once. During the *PhaseInitiating* event the project or subproject is authorized, and then next project activities can be started. During the *PhasePlanning* a project management plan which contains subsidiary plans (e.g. WBS, schedule, cost, quality plans) is defined. Activities performed during *PhaseExecuting* coordinate people and other resources to carry out the plan. *PhaseControling&Monitoring* overlaps with *PhasePlanning* and *PhaseExecuting* and its activities consist in monitoring and measuring project progress and taking corrective action, when necessary. Finally, *PhaseClosing* formally finishes the phase and is the end of the corresponding project or subproject.

9.2. ROLES

In the PMBOK *Stakeholders* are referred as persons or organizations actively involved in the project or whose interests may affected by the project execution or completion. There are two groups of stakeholders in the PMBOK those explicitly engaged in project realization *ProjectTeam,* and others (*Customer, Seller* or *Sponsor).*

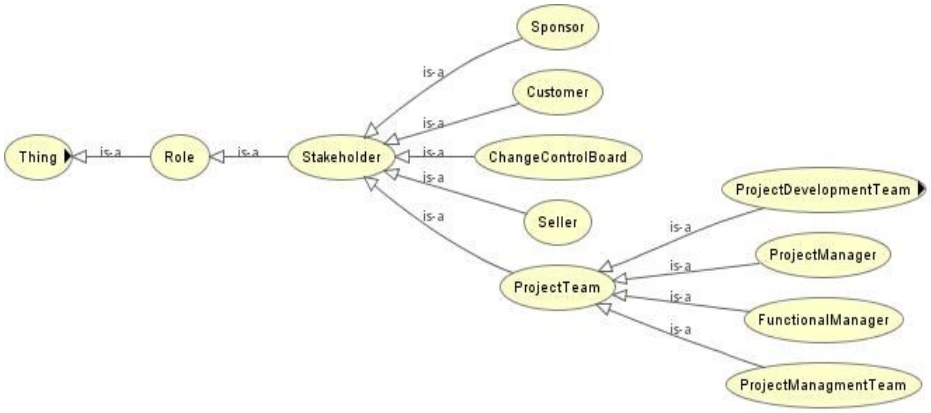

Fig. 5. Roles in the PMBOK

A key person in each project is *ProjectManager*. He is responsible for:

– Developing the project management plan and all related components plans.

- Managing the current project activities according to the schedule and the budget.
- Providing deliverables to customer.
- Identifying, monitoring and mitigating risks.

One of project roles belonging to *ProjectManagmentTeam* is *FunctionalManager*. *FunctionalManager* plays a managerial role within an administrative or functional area of the business concerning human resources, finance, accounting or procurement. Among the stakeholders also appears in the class, which does not represent a single person. It is a *ChangeControlBoard*. *ChangeControlBoard* is a group of people who assess changes proposed in *Monitoring&Control* phase for approval. The board includes *ProjectManager*, *Sponsor* or *TeamMember*.

9.3. ARTIFACTS

In the PMBOK ontology *Artifact* contains all inputs and outputs of the processes listed in the methodology specification [2] and also subsidiary classes used to define their internal structure. There are two main *Artifact* categories in the PMBOK ontology: *Deliverable* and *ManagmentDocument*. Class *Deliverable* define products, services and documents delivered to the end-customer. Class *ManagmentDocument* represents artifacts arising as results of management processes. They are contracts, documents, reports, plans and estimations used for project planning and developing.

One of the most important artifact of PMBOK methodology is *ProjectCharter*. *ProjectCharter* is a document which formally authorizes and initiates a project or phase. It is the only output of *DevelopProjectCharter* process. It consists of project description (modeled by the class *ProjectDescription*), high-level requirements (*ProjectRequirement*), *ProjectManager* assignement and his authority (*AssignedAuthorityLevel*) and lists of important stages that are planned for the project: *SummaryMilestoneSchedule.*

Description of artifacts is supplemented by relations linking them to roles. These relations are: *source* (a source of artifact) and *hasParticipant* (indicating a role responsible for maintenance and updating). An example is the relation: *ProjectCharter hasParticipant exactly 1 Sponsor*.

9.4. PROCESSES

The PMBOK is a process oriented methodology. The standard [7] defines 42 processes classified by their adhesion to ProcessGroups and KnowledgeArea. In the PMBOK ontology each process is described by several properties: input artifacts (the relation *hasInput*), output artifacts (the relation *hasOutput*), roles, i.e. agents responsible for or participating in the process execution (the relation *isExecutedBy*), process group to which a process belongs (the relation *belongsToProcesGroup*), knowledge area (the relation *belongsToBodyof Knowledge*) and tool or techniques that can be applied (the relation *hasToolsAndTechnique*). Each process in PMBOK is associated to the only one Process Group and to the only one Knowledge Area. That constraint is modeled by making disjoint subclasses of Process Group and independently subclasses of Knowledge Area.

10. ARCHITECTURE OF THE SCRUM ONTOLOGY

The Scrum ontology contains four main groups of classes: *Artifacts* (objects that are produced and consumed by project management processes), *Roles* (classifying persons or groups of persons according to their functions), *Events* (time related concepts such as planning horizons) and *Processes*. This taxonomy is supplemented by subsidiary classes: *Properties* (classes without defined structure) and *Values* (sets of atomic values).

10.1. ARTIFACTS

The category *Artifact* contains classes of objects that are created by project management processes. Artifacts that are specific for the Scrum methodology are subclasses of the class *scrum:Artifact*. While defining artifacts, relations between them were modeled, as well as their links to roles.

Examples of artifacts specific to the Scrum methodology are *ProductBacklog* and *SprintBacklog*. *ProductBacklog* contains several elements of type *ProductBacklogItem* specifying requirements (*ProductRequirement*) referring to a certain *ProductFeature*. Elements *ProductBacklogItem* are attributed with business values (*EstimatedBuisnessValue*). *ProductBacklog* contains also *ReleasePlan* (the plan of software development with several milestones equivalent to external releases). *SprintBacklog* contains items of type of *TaskDescription* specifying tasks, their assignment to concrete team members responsible for their execution, their state and estimated time of realization.

Description of artifacts is supplemented by relations linking them to roles. Those relations are: *originator* (the source of artifact), *creator* (the role responsible for creation of artifact) and *isManagedBy* (indicating a role responsible for maintenance and updating). An example of a relation is: *ProductBacklog isManagedBy exactly 1 ProductOwner*.

10.2. ROLES

Basic roles appearing in the description of Scrum methodology are *Stakeholder*, *ProductOwner, ScrumMaster* (team leader), *Team* and *TeamMember*.

Stakeholders are people outside the project team involved or dependent on the course of project work. Typical stakeholders are an investor, business executives, customers and others.

Main tasks of *ProductOwner* are requirements management, their prioritization according to the order determined by business value and controlling requirements doneness. *ProductOwner* is responsible for achieving goals by the *Team*. The responsibility is modeled by a participation of the role in further described processes. Team is self-managing and self-organizing group and its members (*TeamMember*) are responsible for requirements implementation. *ScrumMaster* is looking after the team and its morale, he supervises the work and the compliance of the project practices to the methodology recommendations. The model takes into account a requirement that is essential for the Scrum methodology: team members should be committed to the project execution and should not participate in other works. It was expressed by the inheritance after the class

CommittedTeamMember disjoint with *InvolvedTeamMember*. On the other hand, Stakeholders should be only involved in the project realization.

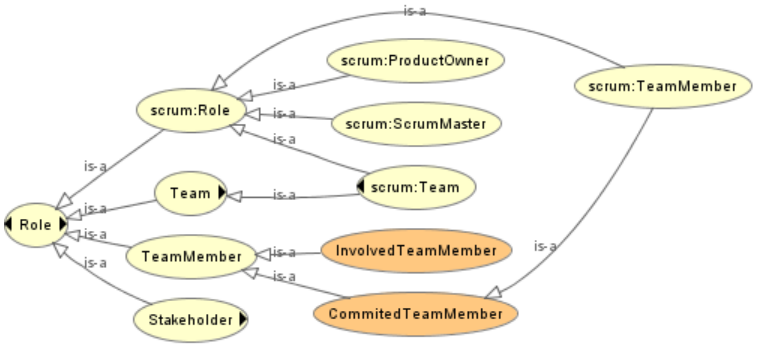

Fig. 6. Roles in the Scrum methodology

10.3. EVENTS

Similarly to the PMBOK model, the Scrum ontology uses the upper ontology of events described in section 9.1.

Classification of events in Scrum methodology is based on the Mike Cohn's description, who presented the agile development on the onion diagram [23]. The layers of the onion correspond to time horizons and form a containment hierarchy. This diagram was later modified by the VersionOne company [24] in order to better adhere to general practices of agile development and this version gained acceptance of Mike Cohn. Planning horizons identified by VersionOne were selected as the basis of events model in the Scrum ontology.

The set of events corresponding to planning horizons comprise:

- *StrategyPlanningHorizon* –planning of the product development
- *ReleasePlanningHorizon* – planning of external releases
- *Sprint* – an iteration, usually lasting 1 month
- *Daily* – daily execution
- *Continuous* – continuous execution

The *Event* category contains also four events describing meetings: *SprintPlanningMeeting* (planning the iteration scope), *SprintReviewMeeting* (review of iteration results), *SprintRetrospectiveMeeting* (lessons learned and experience from the previous iteration, recommendations for the future) and *DailyScrumMeeting* (analysis of progress, impediments and formulating new tasks). Events are linked with temporal relations (Fig. 7.).

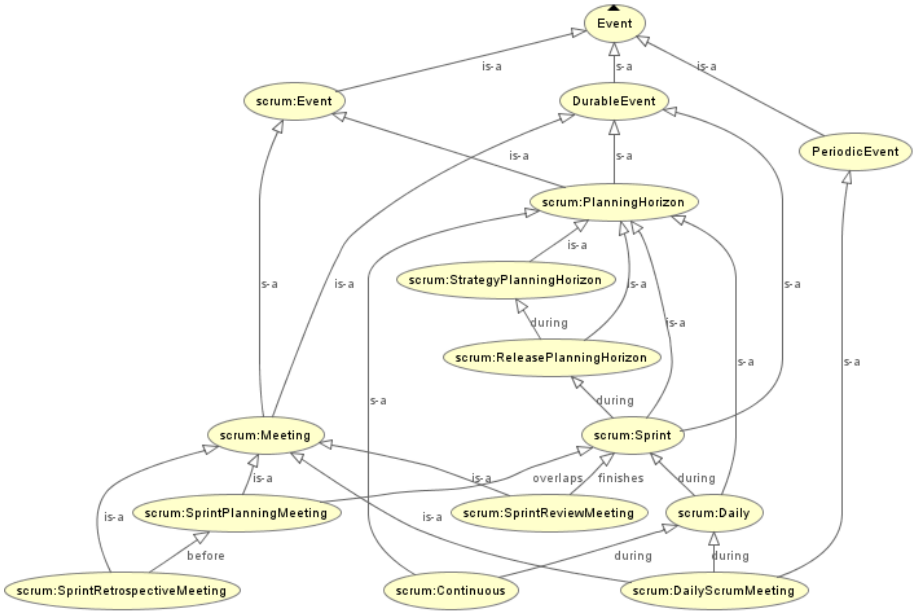

Fig. 7. Ontology of events in the Scrum methodology

10.4. PROCESSES

Introduction of processes into the model of the Scrum methodology was considered as a very important goal. The process description is now a basic tool for building formal models of business activities. Identification of processes in the Scrum methodology should be treated as an attempt to find common conceptual base for potential users of this methodology, but it also can be used as a foundation for further comparison and mapping to classical methodologies represented by PMBOK [7] explicitly enumerating processes.

In the Scrum ontology, similarly to the PMBOK ontology, each process is described by four properties: input artifacts (the relation *hasInput*), output artifacts (the relation *hasOutput*), roles, i.e. agents responsible for or participating in the process execution (the relation *isExecutedBy*) and events, during which the processes are executed (the relation *during)*.

In the Scrum ontology 13 processes were classified and assigned to the subsequent planning horizons:

- In the *StrategyPlanningHorizon* phase the processes *VisionCreation* and *ProductRoadMapCreation* are performed.
- During a *ReleasePlanningHorizon* the following processes are executed: *ReleasePlanning*, *InitialBacklogCreation* and *BacklogUpdating*.
- The Sprint phase encompasses the planning process *InitialSprintPlanning* and the closing processes: *SprintReview* and *SprintRetrospective*.

– During a Daily planning horizon there are executed: control process (*SprintTasksControl*), introduction of changes (*SprintTasksUpdating*), reporting (*ImpedmentsReporting*) and execution (*FeatureDevelopment*).

The described set of processes together with assigned roles reflects defined Scrum practices and may constitute a good foundation for constructing mappings to processes of the PMBOK methodology. It may be difficult to find bijective mappings between processes, in most cases there can be stated an overlapping of concepts. A typical example is partial overlapping of the PMBOK process *PerformIntegratedChangeControl* with Scrum processes *BacklogUpdating* (regarding a change of requirements and the project scope) and *SprintTasksUpdating* (a change of tasks and shedule).

11. CONCLUSIONS

For many IT enterprises realization of projects is the primary mode of operation. From the perspective of managerial staff, the choice between classical and agile project management methodologies can be considered as a critical issue that can influence efficiency, costs, time to market and other factors determining the enterprise competitiveness. The typical scenario is that after a few initial steps that incorporate the formulating the project vision and overall requirements a member of managerial staff should make a decision whether subsequent project activities should be performed according to the framework of classical or agile project management methodologies. The decision should be based on the assessment of such elements, as the project scope and size, the current state of organizational assets, the experience gained from the previous projects, tools, developed software, etc. This implicate that in many practical situations practices of classical and agile project management methodologies can be mixed and adapted to the enterprise conditions.

The suggested approach to integration and adaptation of existing standards consists in using ontological models as descriptions of project management methodologies and conducting the integration by building mappings between these models. The advantage of the proposed solution is the use of a formal language of specification eliminating ambiguities in interpretation of concepts and an opportunity to use proven tools and techniques of modeling and concept matching.

In this work ontologies are used to describe elements of project management methodologies: roles, artifacts, events, tools and techniques. The presented ontologies constitute first propositions of models that will be further developed and refined. An important issue is the problem of ontology modularization and selection of available upper ontologies, e.g. [25] or construction of a set of dedicated upper ontologies adapted to the problem domain.

The next important issue is the validation and evaluation of the quality of ontological models [26]. While building the presented ontologies the OntoClean [27] recommendations were applied. The OntoClean approach distinguishes three properties of classes and their relations that should be preserved in ontological hierarchies: *identity*, *rigidity* and *unity*. Consistent application of OntoClean recommendations assures formal correctness of the model, but can give no guarantee that the model properly describes the selected domain;

therefore it is planned perform the validation of constructed ontologies by presenting them to external experts, e.g. connected to PMI organization.

12. REFERENCES

[1] Marek Jaślan, *Branża IT ma się dobrze*, www.msipolska.pl/rynek_200702.php4?num=445, 2007

[2] Tomasz Sańpruch, www.prnews.pl, Bankier.pl, 2007

[3] Tomasz Szetyński, *Rynek IT rośnie powoli - firmy, które chcą istnieć na rynku zmieniają swoją strategie działania*, www.globaleconomy.pl/content/view/80/17/, 2010

[4] The Standish Group, www.standishgroup.com

[5] J. Laurenz Eveleens, Chris Verhofer: *The rise and Fall of the Chaos Report Figures.* IEEESoftware, 2010, pp. 30- 36

[6] Barry Boehm, Richard Turner, *Balancing Agility and Discipline: A Guide for the Perplexed*, Pearson Education, 204, p. 266

[7] *A Guide to the Project Management Body of Knowledge Fourth Edition (PMBOK® Guide)*, Approved American National Standard ANSI/PMI 99-001-2008, PMI 2008, s. 460.

[8] *The state of Agile Development, 3rd annual Survey:2008*, Full Data Report Conducted June-July 2008. VersionOne Inc. www.VersionOne.com/acnewsletter, s. 18.

[9] Werewka J., *Scaling Management of IT Projects Depending on their Size and Complexity*, 1st CEE Symposium on Business Informatics, ACS, Vienna 2009, s. 181-190.

[10] Mobilizing HP, *Project Management as an Executive Priority. Benchmark Implementation: Primavera, Case Study*, 2004 Benchmarking Partners, pp. 9

[11] Schwaber K., Beedle M: *Agile software development with Scrum*. Upper Saddle River, N.J., Prentice Hall, 2002

[12] Franke, U.; Hook, D.; Konig, J.; Lagerstrom, R.; Narman, P.; Ullberg, J.; Gustafsson, P.; Ekstedt, M.; *EAF2- A Framework for Categorizing Enterprise Architecture Frameworks*, ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing, 2009. SNPD '09, pp.327 – 332

[13] Gruber T.: *A Translation Approach to Portable Ontology Specifications*. Academic Press Ltd., 1993

[14] Wikipedia: *Ontology (information science)*, http://en.wikipedia.org/wiki/Ontology_(information_science)

[15] Rumbaugh J., Jacobson I., Booch G. *Unified Modeling Language Reference Manual, The (2nd Edition)*, Pearson Higher Education, 2004

[16] Protégé: http://protege.stanford.edu/

[17] Jena - *A Semantic Web Framework for Java*, http://jena.sourceforge.net/

[18] Euzenat J., Shvaiko P., *Ontology matching*, Springer Verlag 2007

[19] Pinto, H. S., Gómez-Pérez, A., Martis, J. P. Some issues on ontology integration. Proceeding of the IJCAI-99 workshop on Ontologies and Problem-Solving, Stockholm, Sweden, 1999, s. 7.1–7.12.

[20] Sowa, J. F. Principles of ontology, onto-std mailing list, Wed, 3 Dec 1997, http://www-ksl.stanford.edu/onto-std/mailarchive/0136.html.

[21] Kaneiwa K., Iwazume M., Fukuda K.: An Upper Ontology for Event Classifications and Relations. Australian Conference on Artificial Intelligence 2007: 394-403

[22] Allen J. F,. Ferguson G.: Actions and events in interval temporal logic. Technical Report TR521, Computer Science Department, University of Rochester, Rochester, New York 14627, July 1994.

[23] Cohn M.: Agile Estimating and Planning, Prentice Hall/PTR, 2006, s.330

[24] Agile Development Poster, http://pm.versionone.com/AgilePoster.html

[25] Ivan Terziev, Atanas Kiryakov, Dimitar Mano, Base upper-level ontology (BULO) Guidance, EU-IST Project IST-2003-506826 SEKT

[26] Daniel L. Moody, Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions, Data & Knowledge Engineering 55 (2005) 243–276

[27] Guarino, Nicola, Chris Welty, Identity, Unity, and Individuation: Towards a Formal Toolkit for Ontological Analysis, W. Horn, ed., Proceedings of ECAI-2000: The European Conference on Artificial Intelligence. Amsterdam:IOS Press. Pp. 219-223. August, 2000.