Pere Tumbas
Predrag Matković
University of Novi Sad
Faculty of Economics Subotica
Serbia

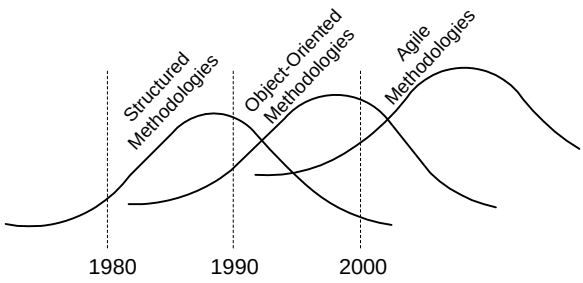
Agile vs Traditional Methodologies in Developing Information Systems

Summary
After the review of principles and concepts of structural and object-oriented development of information systems, the work points to the elements of agile approaches and gives short description of some selected agile methodologies. After these reviews, their comparision according to criteria is done. The first criterion reviews the volume of methodology in which project management is used in developing information systems. The second criterion shows if the processes, defined by methodology, cover appropriate phase of the life cycle. The last criterion shows if methodology iniciates the use of skills and tools in the life cycle phases of developing information systems. Finally, the work compares, according to the key elements of development, traditional (structural and object) methodologies with agile methodologies.

Key words
agile development, information system, traditional methodology

1. Introductory remarks

To the end of the 20th century, two basic groups of methodologies were used, with more or less intensity, in developing information systems – structural and object methodologies.



The structural approach and all structural methodologies are characterized by the flow of in advances strictly defined developmental activities successively done from the identification of requirements, analyses, and designs to the implementation and maintenance. The limits of some activities and phases in development are sharply set. Activities have strictly defined inputs and outputs where the suppositions of realizing next activities are the results realized in the previous activities. The built information system is the consequence of all the activities defined in the selected methodology without exception. This approach has been mostly applied in practice up to now and with regard to this, the number of methodologies being applied is still the most numerous. The sequential flows of activities is recognizable with all methodologies because they basically possess the model of the development life cycle, taking into consideration that in some modifications of this model, the course of activities advances by introducing iterations between some activities. Activities are done blindfolded in the defined sequence with no remainder.

The object approach and object methodologies of developing information systems are based on the "object" paradigm that is used in the course of the bigger part of the life cycle. The evolutionary process represents the base of approaching and developed methodologies that provide a firm connection between different and distant activities in the life cycle of developing an information system. This approach and all the object methodologies are based on the iterative and incremental development of information systems, i.e. on the software development realized through small steps, i.e. iterations. By Any iteration ads, or increments, to the information system, a new or the existing rebuilt functionality of the system. By iteration the information system is added or incremented new system functionality or the existing system functionality is rebuilt.

As projects of developing information system are often late, the exceeding of budgets and time limits in realizing projects, permanent development of technology complexity, constant changes of the user's requirements have brought to the great turn in the realization of methodologies. At the end of the 20th century, besides the cited methodologies, often called traditional or formal ones, agile methodologies in developing information systems appeared. They were built based on the following thinking:

- Development of information systems is a creative work where design activities are dominant,

- Development processes have to be flexible to enable users to change frequently their requirements without consequences,
- Characteristics of individuals participating in development exert the primary influence on the quality of project activities, where the optimal effect is reached by the teamwork.

2. The Agile Approach and Agile Methodologies

The agile approach has faced to the basic problem of contemporary and fast development of information systems. The dominant idea is that development teams can be more efficient in realizing changes if the time can be shortened and the costs of changes of information between the persons participating in development can be lowered. That must be done in the way that will shorten the time limit as the decision is made to the information about the consequence of the decision.

The basic and primary values of the agile approach are:

■ Individuality and interaction versus standard proc esses and tools

The previous approaches in developing information systems paid attention to the processes, considering all the participants in development replaceable components of development. However, the agile approach means that the success of development depends on the project team only, i.e. there is no good software without good "players". These good players are not only the first-class planning engineers and programmers but also the persons with average knowledge and capable to work in the team. Communication and interaction in the team are more important factors for success than the talent itself. The team with average knowledge workers has better hope of success than the team consisting of persons of above average knowledge but who are not capable to work in the team. Development tools, according to the agile approach, have big importance for the success of the project, but the abundance of tools is as negative as the shortage of them. This approach also suggests the use of free tools, not purchasing the license of the newest and the most expensive CASE products. Taking up better tools is done when the cheapest and used ones do not satisfy the needs of development.

■ Practical software versus voluminous documentations

The followers of the agile approach think that voluminous documentation in unnecessary and that it makes negative effects. Therefore, to their opinions, it is better to spend time in carrying out the program code. In fact, the code is not the means of communications, but documentation should be rationalized for the needs of decision-making and training new members of the team. Short, rational documents do not limit their training because, to the agile approach, it is done in cooperation and within the framework of the team. The new members gradually become parts of the future teams by interaction with the old members of the team.

■ Cooperation with users versus contracting party relationship

The agile approach means the active and constant participation of users in the development team, during not only the identification and expressing their requirements. The periodical participation in development leads to the failure and low quality in realizing the project of information system. To make the project successful, the authors of agile methodologies base the whole procedure and the time of development on the frequent contacts with their users who give initial or recurring information. According to their opinion, such a relationship is better than a contracting relationship where the cooperation of contracting and polarized parties is explicitly cited.

■ Change adapting versus keeping up with the plans

The capability for frequent reaction on changes represents the decisive component for success or failure of the project. The agile approach suggests that we should pay attention to carrying out the projects. They should be flexible and adaptable to changes because the directions of project development cannot be predicted for the far future. The business environments change constantly, as well as the users themselves who change their requirements in relation to the system that is functioning. The strategy of planning, as it is suggested, means a detailed plan for the first several project weeks. It is a rough, approximate plan for several months and a simplified plan for the time after that. Namely, tasks for several weeks must be well known, for the next several months only roughly. As for tasks or ideas after a year, we should vaguely catch them in sight.

All these cited values can be realized through the following twelve principles, contained in the Agile Manifesto:

- Satisfaction of users has the highest priority by an early and frequent delivery of usable software. Its use in the earliest phases of development and its frequent distribution bring several benefits.

First, the development team has recurring information about everything that is already done, i.e. if the way of development functions properly. In addition, users are able to use the realized solution very early if they appraise that the built functionality is proper and enough or just to look at it and require its further changes.

▪ Changes of requirements are always welcome, even at the end of development. With frequent changes, the realized solution adapts to the user's changed needs and it contributes to the higher quality. Agile development teams tend to keep the structure of solution very flexible so the changes of requirements exert little influence on the software.

▪ Deliver frequently usable solutions to the user, in intervals from several weeks to several months, giving preference to shorter periods. The basic aim is to deliver solutions earlier and oftener. By these short intervals, we get frequent recurring information from the user based on which, if necessary, we could correct mistakes of the delivered solutions. Not all deliveries, until the last one, contain documentations.

▪ Users and planning engineers work together every day until the end of the project. To have an agile project, there must be important and frequent interactions between users, development teams and all the other interested parties in development.

▪ Motivated individuals have to work and perform tasks in the project. It is necessary to provide them appropriate support, working environment and they must have confidence in themselves that the work will be done in time. Special characteristics the individuals must have are kindness, communicativeness and talent. In the agile approach, the participants of development represent the most important factor of success. All the other factors as processes, standards, the environment and management are of the secondary significance and they can be always changed during the development. In addition, if some activities or methodological steps of development represent the obstacle in realizing the tasks, they can be changed.

▪ The criterion of advancing in the project is the degree of software usability. The progress of the agile project in developing information systems is measured by the quantity of software solutions that are used, not based on the phase of development of the project, the quantity of the written code, quantity of created documentation, and similar.

▪ The agile process supports the sustainable development. According to the agile approach, the development team must keep lasting and constant strength and speed in development. The tired and stressed staff is not desirable in the project because they do not attain good results. Therefore, all the participants in the project team must have reasonable working engagement that enables them to remain constantly employed and calm. The agile team independently determines its tempo, taking into consideration not to spend the tomorrow's energy in order to do today more.

▪ Constant orientation to technical perfection and good design increases agility. If the system design is concise and well settled, it is easier to change it so it also increases agility. The obligation of the planning engineers is to give good designs and to consider and update them during the whole development.

▪ The base of development is simplicity, i.e. the capability to maximize work that is not done. The agile approach insists on implementation of only agreed characteristics. The prediction of future problems is not its characteristic, but solving the current problems. The basic idea is to perform work, to look for a simple and qualitative solution in order to change and adapt it when the problem appears.

▪ The best architecture, requirements and designs bring to the self-organizing team. Responsibilities and roles are in the team, not out of it. They are given the whole team and in the team, they are distributed as agreed. The members of the team work together in all the activities of the project. There are not single responsibilities for architecture, requirements or testing, but the whole team bears them.

▪ The way of realization of better efficiency is controlled in regular time intervals, and according to it, the behavior is adapted. The development team meets regularly to analyze and discuss the work of every member of the team, with a view of increasing his or her agility. Therefore, its organization, rules, conversation, connections change periodically in order to keep agility.

The term agile is selected by the group of persons with great experience in developing information systems. The starting suppositions were that the process of developing software and information systems is necessary to the turbulent business and technological environments, the process that initiates changes but quickly responds them. In the same time, the process must include responsible participants and their good organization. Special attention is paid to participants, their talents, skills and capabilities. Orientation to participants is the

most important characteristic of agile methodologies. The complete process of development is adapted to individuals.

In agile development teams, competencies of individuals represent the critical factor of the project successfulness. To agile methodologies, if the individuals in the project are of high quality, then they can realize the expected goal by any applied process of development. In contrast, there is no process of development that can replace their incompetence. In the same time, the lack of the user's support can easily destroy the project of development, as the inadequate support can prevent the end of the project.

Although formal methodologies did not neglect the capabilities of individuals, they still considered them in the other way, as well as they developed them in the other way. Strict and standardized processes, known and previously applied methodologies of development, were designed to standardize and adapt individuals according to the organization of the process, while agile processes emphasize unique capabilities of individuals and teams. Namely, processes cannot bridge the lacks of competence of individuals. Teams are self-organized, with intensive communications in the framework and out of the organized limits. They can immediately change their structure in order to adapt to changes.

Agility means that the team has a mutual aim, mutual confidence and respect, mutual and fast treatment in decision-making and the capability to solve all ambiguities. The agile team working in the framework of the rigid organization has difficulties, as any individual working in the rigid team. In these teams, cooperation is dominant at all the levels of management. It is not important who makes decisions, but cooperation and providing information for decision-making. The persons of different capabilities, skills and talents take part in the project of development, the persons that work in the close physical environment and who respect the organizational culture. Persons, the environment and culture are in a strict interrelation.

The agile approach and agile methodologies are not appropriate for every situation in development. The intrusion of agile principles the process-oriented and non-cooperative organizations does not have success. The intrusion of exceptional changeable process to calm and cool teams leads to the team disintegration. In addition, agile development is difficulty carried out in the teams with a great number of members. In the agile development, the teams with nine members show most success. The agile development was successful in extreme, complex projects and the projects with many changes. The environment where this approach reaches the best results is the organizational culture oriented to people and cooperation.

Among the most known and the most frequent applied agile methodologies are:

- Extreme programming (XP),
- Scrum,
- Crystal Methods,
- FDD (Feature-driven development),
- DSDM (Dynamic Systems Development Method),
- ASD (Adaptive Software Development),
- Lean development and some others.

Extreme programming (eXtreme Programming – XP) represent the most known agile methodology. Its creator is Beck Kent (1999) who created it with the basic wish to move the limit of expected performances of the system and to reject the majority of safety systems present in the other methodologies. The XP methodology consists of several rules and a small number of procedures easy and simple to use. The software is developed by iteration, usually lasting two weeks, when the users' stories are implemented, i.e. the software characteristics that are mutually formulated by the future users and planning engineers.

The basis of XP is the cooperation with users and the strong feedback; therefore, users are included in planning and the whole development. The user orients the whole development team and the new software is placed at his disposal every couple of days. The emphasis in development is placed on testing by the users' tests, programming in couples and testing-led development in order to get the high-quality code. XP is a combination of simple practice that emphasizes communication, teamwork, requirements of the user and his/her satisfaction. According to the creator of this methodology, its cycle includes the following phases: from research, planning and iteration to distribution, production, maintenance and abandonment.

Scrum is an agile methodology developed by Ken Schwaber and Jeff Sutherland with the late cooperation of Mike Beedle. Scrum provides the framework for project management and aims to advance work productivity of small teams to 10 members that were previously limited by hard and the process-oriented methodologies. The Scrum methodology neither requires the use nor provides specific techniques fro software developing. Instead, it requires some management capabilities of the team members. The process of information

system development is structured into three phases: pre-play, development and post-play.

The development activities of this methodology are characterized, before all, by the activity of defining the Product Backlog List) that includes all the identified user's requirements disregarding if they derive from the user, manager or the software engineer. This list constantly changes and updates by new and detailed items. Besides, the system development is done in Sprint that represents the interactive cycle during which the system functionality is advanced. Every Sprint includes all the phases of the life cycle of information system development, where its average phase of development lasts from one to four weeks. After every Sprint, the demonstration of functionality is done, the attained results are checked and new tasks for the next cycle of Sprint are defined based on the changes in the environment.

Crystal Methods – The creator of the family of crystal methodologies is Alistair Cockburn who gave the names of some methodologies based on colors the crystal emits (crystal clear, yellow, orange, brown, blue and violet). Instead to the standardized processes, these methodologies are oriented to the teams that are the carriers of information system development, communication between the members of the team and their satisfaction. The aim of these methodologies is to identify the components important for the success of the project and to provide the development team with enough strength and freedom to perform their job in an amusing and creative way.

The participants of development and their collaboration and cooperation in the project are in the basis of this methodology. For every project, this methodology finds the simplest and the most compact combination of characteristics of the team and the process. As every team has the different set of capabilities and skills, the appropriate process of development should be applied to it. What process will be applied depends on the size of the team and the factors of criticism. The difficulty of some methodology is designated by color. The darker the color, the more difficult the methodology is.

Feature Driven Development is the methodology developed by Jeff De Luca and Peter Coad. It is developed for the needs of big projects of information system development. The methodology includes five processes for realizing the development: global model development, building list of functions, planning according to functions, design according to functions and building according to functions. The development is divided into small functional blocks having some value that are called Features. The first three processes are performed in the successive line, while the last two repeat, i.e. enable iteration. Most of time and efforts in realizing the project is spent in the last two processes.

The FDD methodology includes the set of the"best procedures". The creators of methodologies think that if they are not new, their specific combination of five FDD processes give a unique and specific approach to every project.

DSDM is the methodology derived from the practice of the fast development of applications. The basic idea of this methodology, created by Jennifer Stapleton, is that, instead of determining the functionality of information system and adapting the time and resources in order to reach that function, to determine the time and resources in order to adapt to them the functionality of the system. Determining the time of delivery and limiting resources is to establish the process that completely satisfies the user's requirements.

To the philosophy of this methodology, noting can be perfectly done at first attempt so the development of information systems should be considered as a research project. DSDM is a development process that includes three phases: feasibility study, business study, iterative functional model, iterative design, and building and implementation. These phases are not strictly defined and they should not be performed. Every one has several key tasks that can be modified if necessary. The first two are sequential and are done once; while during the other three phases the development is iterative and incremental.

3. Comparison of Agile Methodologies

The supposition for the choice of an appropriate agile methodology is to know well its comparative advantages in relation to all the available alternatives. The simplest way to make decision is to do it by the analysis of reports generated and based on the other experiences in their applications. However, researches show that there are not a critical number of such reports, connected to agile methodologies, in order to compare them. Therefore, the need to compare them scientifically is evident.

Any kind of comparing agile methodologies, without traditional-formal methods, is extremely susceptible to subjectivity. Introducing the quasi-formal approach of comparison, the problems caused by subjectivity are prevented, the problems appearing with the non-formal approach. It is possible to establish the quasi-formal method of comparison in many ways:

1. Describing conditionally ideal methodologies, and then by comparing and evaluating the selected methodologies relating to it.
2. Identifying the set of basic characteristics deduced from the set of known methodologies, and then by comparing every methodology with the identified set.
3. Formulating a priory hypothesis about the requirements connected to the methodology. Then, it is necessary to test the formulated hypothesis by practical evidence from comparative methodologies.

Some of agile methodologies have been concisely represented in the previous text. A general conclusion is that they are based on the same or similar concepts; still the methods used are very different. Therefore, there are similarities and significant differences, too. To compare them, the following criteria are the most often used in literature:

a. Methodologies supporting activities of project development management,
b. Methodologies supporting the phases of the software life cycles,
c. Level of concreteness/abstractness these activities have, on which these methodologies are based and how they use some resources,
d. Level of adaptability in the concrete situations of applications,
e. Level of empirical foundation.

Figure 1 shows the graphic illustration of five compared methodologies according to the first three cited criteria. Every methodology is represented in tree blocks. The upper block shows the volume of activity of project management that the observed methodology includes. The middle block shows if the activities, defined by the methodology, cover the appropriate phase of the life cycle. The lower block shows the level of concreteness of activities and resources of the methodology. The gray color on the graph expresses the presence, while the white color represents the shortage of some features.

Analyzing this figure, we can conclude that agile methodologies are focused on different phases and activities of the life cycle in developing information systems. DSAM is the only one, of all the compared methodologies, that supports all the activities of the life cycle development. Some methodologies are focused more on the activities of project management (Scrum). Analyzing the completeness of compared methodologies, and taking into consideration its vertical and horizontal dimension, i.e. the completeness and representation of some phases, we should emphasize that no one of these methodologies is too detailed or too precise.

In addition, the activities of project management, providing the tasks in development be developed regularly, are differently represented in some methodologies. FDD is a methodology that completely supports project management, while X, as the most often used methodology of the agile development, gives only partial support. In selecting methodologies, the number of the development team represents one of the basic criteria. Some methodologies (XP, Scrum) are focused on small teams, less than ten members. The other methodologies (Crystal, FDD, and DSDM) plan bigger development teams. Of course, bigger development teams increase the need to documentation; this makes methodology less agile. Considering the compared methodologies, we can conclude that DSDM is different from the others, and so it lines up into less agile methodologies.

At last, we should emphasize that abstractness prevails in the majority of agile methodologies, i.e. there are no instructions for their use with detailed explanations of practice and activities that should be taken, as well as there are no explanations how the tasks are performed. Just some methodologies and in some phases give detailed instructions.

Agile methodologies can be compared in many more ways. One of the possible referential frameworks for comparison is to establish four potential quadrants, based on the following characteristics set on the coordinate axes:

- Level of documentation. The level of documentation is represented on the horizontal axis, as well as the existence of formalities in the observed methodology. i.e. the existence of completely defined instructions and rules;
- Sequential/Iterative approach. The relationship of the sequential approach (linear approach, with integration and testing in the later phases of development and with the high level of risk) and iterative approach (the approach oriented to the risk minimization, with continual integration and testing and with development in the rows of iterations) is represented on the vertical axis.

The review of this referential framework for comparing agile methodologies is illustrated in Fig. 2.

The analysis of the preciously described characteristics of compared agile methodologies placed into the referent framework, illustrated in Fig. 2, draws a conclusion that the majority of illustrated

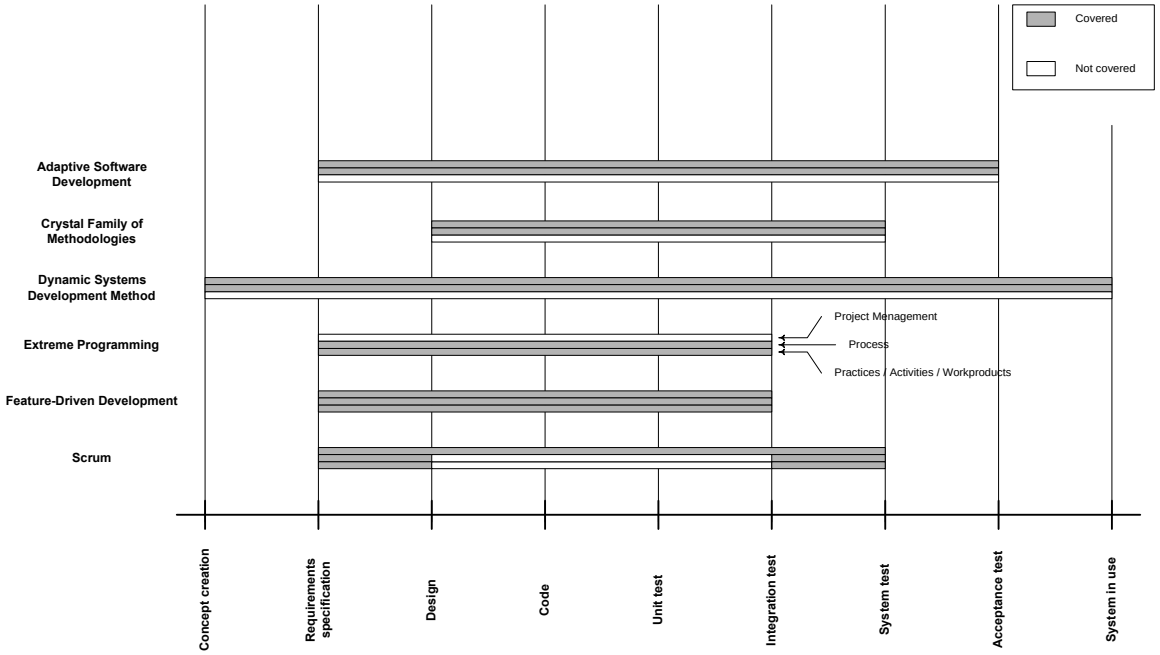


Figure 1. Comparison of agile methodologies

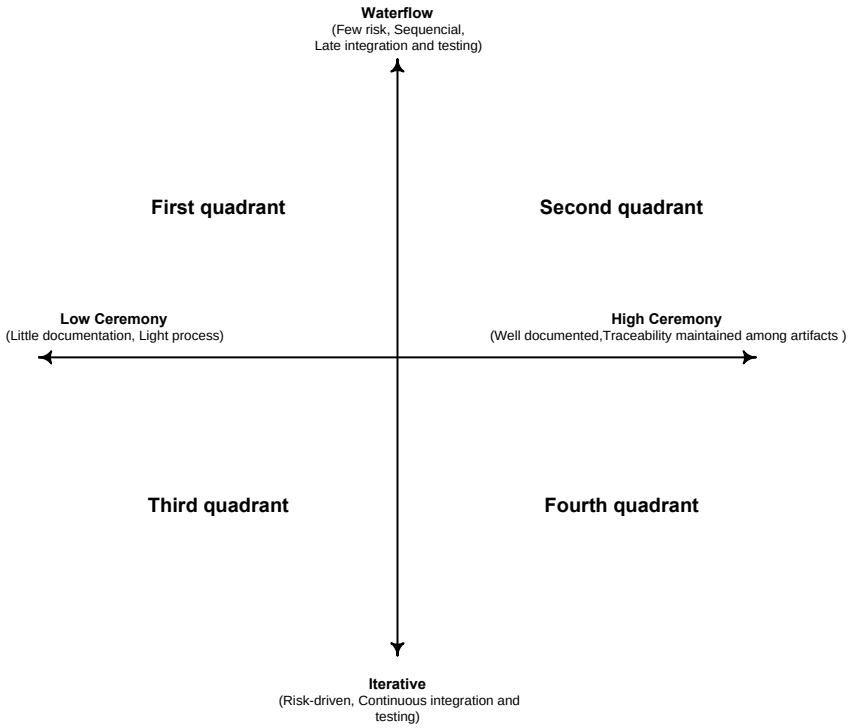


Figure 2. Map for Process Comparison

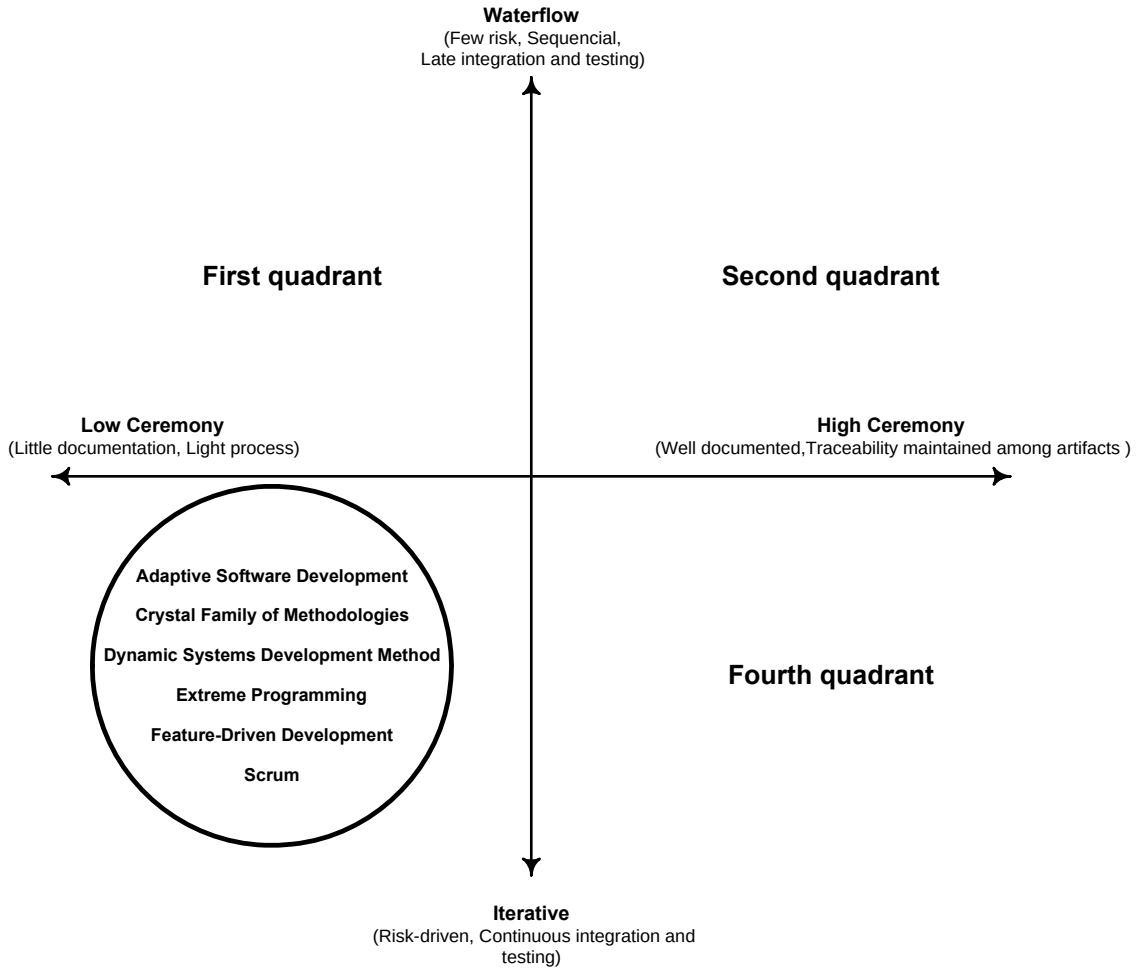


Figure 3. Agile Processes on the Process Map

methodologies are in the third quadrant. All the methodologies, found in the third quadrant, use the iterative approach in development. They suggest a minimal use of documentation and formality, and therefore, they are ideal for using in small and non-complex projects. They are the following methodologies: Agile Software Development, Crystal Family of Methodologies, Dynamic Systems Development Method, Extreme Programming, Feature-Driven Development i Scrum. The following review in Fig. 3 illustrate the place of numbered methodologies in the referential framework.

4. Agile and Traditional Methodologies

To compare agile methodologies with traditional methodologies, it is necessary to emphasize that a number of criteria can be defined, depending on the set goal. This work points to some criteria only and gives short comments on them that can be the subject of special further researches.

1. Iterative vs waterfall

Agile methodologies, relating to this criterion, are homogeneously and exclusively determined for the iterative development, while traditional methodologies are based on the sequential development that is only acceptable and normal.

2. Workflow

Agile methodologies in the process of development emphasize the mutual interactivity and dependance of phases of design, implementation and testing, where they must be competetive and iterative. The definition of all the processes must be concise and easy to remember. Traditional metho-

dologies, however, recommend a stric sequence of the cited phases and it cannot be violated. Processes must be documented and generaly defined.

3. Requirement Quality

Agile methodologies pay special attention to the process of defining the users' requirements. Their quality decisively determines the quality of the final solution of development. Therefore, the concise and documented presentation of requirements is emphasized. Users can frequently see, analize and comment the future solutions in the process of development and from the earliest phases. Permenent comments and changes of users' requiremnts are desirable because they contribute to the higher quality of the software solution. Traditional methodologies also insist on the documented requirements, but without all details. Requirements are identified at the beginning of devlopment and they change only exceptionallly. The user can give his comments not untill the end of the developmental cycle, although the visible solution is available to him.

4. Knowledge Transfer

Agile methodologies in transferring knowledge between the participants during development suggest verbal and frequent communications. All relevant information change in everyday and direct contact of the development team. The clear and legible source code of the program solution is the basis of every communication. So that all the participants could understand the system and its functions, traditional methodologies require documentation that should be done directly and immediately in the course of realizing some phases of development. If the generation of documentation is not provided timely, the whole process of development can be endangered.

5. Team Composition and Size

Agile methodologies base their development on small and operative teams, easy changeable, having persons with general knowledge with in advance defined tasks and determined individual resposibilities. In traditional methodologies, development is done by big and organized teams with specialists for some activities in the phases of development.

6. Planning and Monitoring

In realizing the activities of development, agile methodologies make the task lists at the beginning of development that is permanently kept during development. Keeping up with the realization of development, i.e. the performance of activities in the task list, is done in order to see not only the advancement of development but also to control the quality of realized activities and their correction if necessary. Traditional methodologies, in perform-miing planned activities, use the Gant diagrams which include, at the start, defined all the project activities. They are also kept during the project realization, but they are controlled only from the standpoint of performance. The quality of activitieis not controlled during every phase of of development.

7. Managing Change

Managing the changes in agile methodologies is necessary and it is a key component of agility. Changes are necessary and we should not avoid them. In contrast to this, traditional methodologies control the level of changes and they require their minimization.

8. Philosophy

The basic philosophy of agile methodologies is that the process of development is considered as a creative process that can be planned, but the detailed plan is always unrealistic. Success is delivering value for money. In contrast to this, traditional methodologies deal with the process of development as the process of making the future solution, so it is advisable to make a detailed plan of activities. Success is meeting the initial predictions of cost and schedule.

Regarding that traditional methodologies have been present for long in developing information systems, it is not necessary to emphasize the reasons for their name. However, it may be important to emphasize the reasons why agile methodologies are called agile methodologies. They are methodologies where software development is incremental (small software releases, with rapid cycles), cooperative (customers and developers working together with close communication) and adaptive (the method itself is easy to learn and modify, well documented) and adaptive (able to make last moment changes).

References

Literature

1. Abrahamssona, P., Warstab J., Siponenb M., Ronkainena J. (2003), *New Directions on Agile Methods: A Comparative Analysis*, IEEE
2. Everette, R. K. (2002), *Agile Software Development Processes: A Different Approach to Software Design*, Software management
3. Steve Hayes, Martin Andrews (2003), *An Introduction to Agile Methods*, Khatovar Tehnology
4. Alistair Cockburn (2002), *Agile Software Development*, Pearson Education
5. Per Kroll, Philippe Kruchten (2003), *The Rational Unified Process Made Easy*, Addison Wesley,

Web sites:

1. The Agile Alliance // *http://www.agilealliance.org* (referred on 20/09/2006)
2. Crystal and Adaptive Software Development // *http://www.crystalmethodologies.org* (referred on 17/09/2006)
3. Software Development Magazine // *http://www.methodsandtools.com* (referred on 10/09/2006)
4. Ron Jeffries' Agile Development Site // *http://www.xprogramming.com* (referred on 18/09/2006)

Pere Tumbas, Ph.D.
Associate Professor

The Faculty of Economics, Subotica
tel.: +381 24 628 001
e-mail: ptumbas@eccf.su.ac.yu

Pere Tumbas is Associate Professor at the Faculty of Economics in the field of Information Systems and Engineering. In his career, he has taught Information Systems Development, Management Information Systems, Object Software Engineering, ERP Systems, and related subjects at undergraduate and graduate studies. He is the author and coauthor of several textbooks, handbooks and monographs in the field of Informatics, as well as over 120 works presented and published at international and national conferences, symposia and magazines. During more than 25 years of working at the Faculty, he has participated in carrying out a number of scientific and professional projects and studies in his field. He is editor or member of editorial boards in several international magazines, as well as member of several programming boards of scientific meetings and symposia. He is also member of professional organizations and associations: EUNIS (European Universities Information Systems), IEEE and AIS (Association for Information Systems).

Predrag Matković

The Faculty of Economics, Subotica
tel.: +381 24 628 100
e-mail: pedja_m@eccf.su.ac.yu

Predrag Matković worked at the Faculty of Economics since 2001. Currently he works as Technical Head of the Information-Documentation Center, the Faculty of Economics, Subotica. He also teaches Management Information Systems, Object Software Engineering, ERP Systems, and Information Systems Development. The fields of his special interest are methods, techniques and methodologies of information systems development, as well as the implementation of different types of information systems in some spheres of activities. He is the author and coauthor of several works in the field of Business Informatics at international and national symposia and magazines. He is the member of AIS (Association for Information Systems) and Technical Secretary of AIS Branch in Serbia.