Management of Program Projects in Conditions of Unstable Development Teams

Igor Nikolayevich Skopin

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.75874

Abstract

Project management issues of teams with outsourced developers are discussed. Peculiarity of these teams is the difficulty of recruitment and, as a consequence, the instability of staff. We propose a special approach to overcome the issues in these circumstances. It is based on the use of a special role in the project team, called the personnel coordinator, who allows risk mitigating of project failure. The main result of the proposed work is the technique of tuning Highsmith's Adaptive Software Development approach to the conditions of projects developed by the unstable team. Another important result is the definition of the functions of the personnel coordinator, which he/she must fulfill in the project.

Keywords: management of software projects, personnel coordinator, the life cycle of software development, hard and agile methodology, adaptive methodology

1. Introduction

The organization of software projects management has long been the subject of research, the goals of which are to increase the efficiency of collective work and to support the quality of the developed products. Today, this area has developed as an independent discipline, offering various methods of managing software projects (commonly called methodologies), which have proved themselves in the practice of designing software systems. Among them are two classes of methodologies: rigid, also called heavy, or even monumental, and agile (accelerated, lightweight). The first prescribe to rely on the collection of the most complete set of requirements and their careful analysis, resulting in the construction of a seriated often call waterfall or an iterative development of the project (most of such methodologies can be found in [1]).

The latter abandon detailed a priori development planning and focus on the priority implementation of the vital needs of users (see, e.g., [2]).

Studies of problems of project management always pay attention to methods of organizing teamwork. As an illustration, it is appropriate to point out the bestseller written by F. Brooks [3], which shows that the problems associated with the division of labor in the project team and identified in the 1970s of twentieth century remain relevant today. Some specialists, noting that human resources (HR) are the main factor determining the success of project development, offer methods of accounting for it when choosing a methodology [4, 5]. At the same time, there are practically no recommendations on how to act as a manager when he/she does not have the opportunity to work with an established team with sufficient qualifications for the project. The peculiarity of this situation is the instability of the development team and the need for ad hoc design solutions. It is typical for start-up companies trying to organize their business for the implementation of a promising idea, but do not have the resources to form a full-fledged team that can act within the framework of solid methodological support, for example, in the style of the MSF (Microsoft Solution Framework) approach [6]. It is tempting to use the rules for organization of works like MSF, but the problem with most of these methods is that they are applicable only if the projects are executed by working groups of specialists who are not only qualified but also well organized in interacting with each other. This is not only a problem of the staff deficit, because of which it is necessary to employ outside persons. It often turns out to be impossible to ensure a constant workload of employees with the necessary qualifications, to provide them with comfortable working conditions and stability of wages. All this increases the risk of instability of the team.

In accordance with the Capability Maturity Model for Software (SW-CMM) standard, which is supported in numerous approaches to improving the software development processes that go back to the Paulk, Curtis, Chrissis, and Weber ideas (see [7]), instability of the command is usually considered as a characteristic of the initial level of the organization’s maturity. The standard assumes that if an organization pretends to grow in maturity, it should strive to move to the following levels: repeatable, defined, managed, and optimizing by continually improving the design discipline. The standard holds that moving up the ladder of maturity makes the organization resistant to instability by developing not only the target products of the project but also documents reflecting the accepted discipline. In other words, the project team should gradually get rid of the influence of the human factor, which in its essence leads to a rigid development strategy, which all project developers are required to take.

We should note that when the instability of the team is objective, the maturity condition for SW-CMM is not achievable. However, this does not mean the impossibility of successful activity. The most notable example is outsourcing. If a manager gives the development task for an external executor as a closed proposal, if she/he composed this proposal exactly accurately and unequivocally, then the risk from outsourcing reduces to an acceptable level, and certain advantages be achieved. Nevertheless, the failures of unstable teams are possible and frequent, in many respects, they occur by the lack of suitable methodological support. In the proposed chapter, we discuss some fundamental aspects of the rational organization of the production process of unstable teams, the account of which allows the project manager to build a reliable methodology for the project activity.

The proposed material based on observations and analysis of the experience of several unstable teams, who managed to build their activities so that a better part of the projects developed was successful. The most significant contribution to this analysis made the information on the remote team that managed to determine the rules of work with external executors of project assignments during the development of a series of projects. A distinctive feature of this team is the great attention that it pays to training employees [8].

Today, communication tools allow organizing teams of performers working remotely. It has become quite common for them to be executed projects by outside performers, using e-mail, chat rooms, and teleconferences. This leads to teams with employees who may not even know each other. In most cases, remote commands are unstable. It is necessary to provide for special rules of interaction for them. We discovered this aspect of instability by observing the activities of the abovementioned successful teams. It largely influenced the approach of the project management, presented in the following sections.

2. Features of the project team

Unstable teams of project developers can be built in different ways, but always the project executors in such a team are divided into three groups:

1. *Key developers*—a stable part of the team. Its members have their own interest in the effective development of the project and obtaining meaningful results. They do not need external stimulation. Key developers are often the initiators of the project. These are employees, whose exit from the team is highly undesirable for the success of the project.
2. *Constantly working developers*—performers who occupy roles in the project, allowing substitution without significant damage to the project. Their distinguishing feature is that, not being key developers, they are available for constant interaction.
3. *Optional developers involved*—performers who are used to fulfill individual tasks, but not on an ongoing basis. The requirements for qualifications, discipline, and other qualities of this employees involved can be different. With the uniqueness of the qualification of the developer involved , in case of his dismissal, the risk of non-fulfillment of the project task increases. The need for such employees defines the team as unstable.

With constantly working developers, management problems do not usually arise—their reliability level is known. However, if there is a need to assume the optional developer involved, then the manager has to decide to whom and on what conditions it is possible to propose a task for the solution. The main reasons for this situation are as follows:

- Inadequacy of the capabilities of key and constantly working developers for the implementation of the project in an acceptable time.

- There is a lack of qualification of the employees of the first two groups for the implementation of the project, and as a result, it is necessary to search on the side of either the executors of the tasks or for consultations, possibly for training.
- Simultaneous execution of several projects, because of which key and constantly working developers concentrate only on the most important projects.
- There is no need to hire employees for full working hours.
- The need to recruit new constantly working developers, which can be met through well-established optional developers involved when they performed tasks.

The style of work of the project manager of the unstable team assumes a permanent search of outside employees involved to perform autonomous tasks. It is necessary to distinguish tasks that involve obtaining the results needed for the development of the project environment and such tasks, the implementation of which should contribute to the success (in all senses) of project development. The latter may be loosely associated with the project, and decisions about their announcement made, for example, for checking the employee. In any case, the autonomous task should be precise and contain an awareness of when, for what, and how the results obtained would be used in the project. In accordance with the answers to these questions, the project manager should formulate regulations of work both for the performance of autonomous tasks and for products received from third-party developers. It should be noted that the autonomous task must be precise formulated. It should contain pointing of when, for what, and how the results obtained are used in the improvement of project activities. When the formulation of the task assumes repeatable using of its fulfillment results, in addition to the usual work acceptance, a special analysis is required *before* and *after* its setting and execution. The first is planning reusing, and the second is the decision of adaptation the result to the project environment evolution.

Instability of the team most often leads to the fact that it is necessary to allocate resources to adapt the skills of employees to the requirements of the project. Primarily, this concerns constantly working developers who have proved themselves in the performance of assignments as potentially effective participants of the project. Do not neglect the need for training and optional developers involved. Even in cases when they have the skills and experience necessary for the project, the autonomy of their work does not promote the growth of their status to constantly working developer and even more so to a key developer. Activities aimed at, on the one hand, the study of the feature of the project being developed, and on the other hand, the acquisition of common skills and experience, needed to support the process of stabilizing the team. These activities must necessarily be provided for when the project developers seek to become a mature and stable team. At the same time, one cannot ignore that training, as a rule, is very expensive and therefore may not be profitable in the current project situation.

3. Mini-cycles of iterations and the life cycle of project development

Typical for any reasonable methodology, a cycle of project development with an iteratively repetitive sequence of overlapping stages of work, we present as follows (see **Figure 1**). Here and later, we use the notation of the life cycle diagrams proposed in [9]. In this notation, vertical

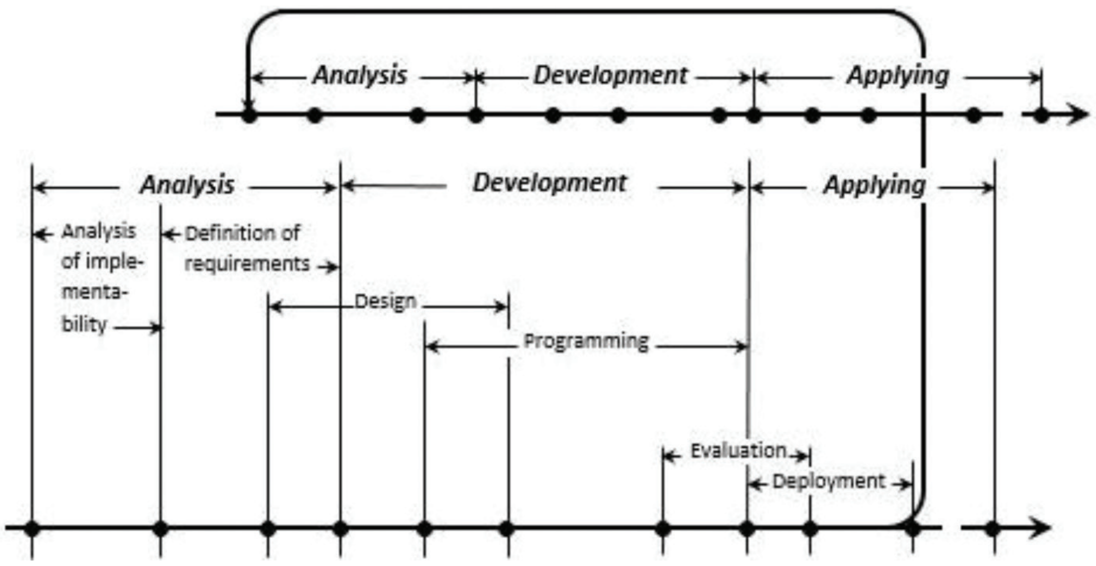

Figure 1. Life cycle of project iterative development.

columns provided with captions show the project phases. The inequality of columns height shows the possibility of overlapping phases: the joint performance of works on a common area of neighboring phases. The bold horizontal arrows present the lifetime of the project iterations. Breaks of arrows indicate that the use of the iteration results continues beyond the development cycle. Crossing the boundaries of phases with lifetime lines are control points and milestones of the project. The colored circles mark them. We display the lifetime of the first iteration with more details than of the second one since their structures are the same. The arrow from the end of the deployment phase of the first iteration leads to the next iteration, possibly with overlapping them. The images of iterations with a shift relative to each other reflect the character of the project development as an ongoing process, during which new tools offered to the user take into account the experience of applying the results of the already passed iterations.

For definiteness, we use the following general breakdown of the process of developing software systems into phases that corresponds to the generally accepted structure of projects (we will use this decomposition later in the discussion of the proposed methodology in Section 5):

1. *Analysis of implementability*—identification of tasks, the implementation of which is relevant for solving user problems that determine the project objectives.
2. *Definition of requirements*, or *development specification*—the development of agreements on what and how should be done in the project.
3. *Design*—definition of the architecture of the system being developed, its decomposition, that is, affirmation of the component structure.
4. *Programming*—creation of software components identified during the previous phase and their verification in terms of meeting the requirements.
5. *Evaluation*—checking the significance of the results obtained both from the point of view of the user's need and for the prospects of the project's development.
6. *Deployment*—transfer of a software product for distribution and use.

This standard life cycle schema for the iterative development process is not entirely suitable for cases where optional developers involved perform some autonomous tasks. Each such task requires the organization of a special process, which we call a mini-cycle, nested in the main iteration of the project to complete the task. The mini-cycle is associated with the splitting of the main process into two branches, performed simultaneously: the continuation of the project and the identification of the task. Splitting can occur at any point in the life cycle of a project when key employees are aware of the need to solve an autonomous task and formulate it for an optional developer involved.

In a mini-cycle (see diagram in **Figure 2**), key developers identify the task for outsourcing, and optional developers involved carry it out. It is clear that for the correct statement of the task, its analysis is necessary, and at the end of the mini-cycle, key developers should make a decision about future using of obtaining results. The latter includes a decision about when the splitting should end, that is, the definition of the latest point in the lifetime line of the iteration, when the results obtained remain useful for the project. The diagram indicates this point as ***F***.

Consider the performance of splitting and mini-cycle details as follows:

- The appearance of splitting occurs at point ***A*** of the lifetime line of the iteration. This is the moment when the key developers realize the need to perform a certain task.
- The mini-cycle begins with an analysis of the identified need at point ***B*** of the lifetime line. Because of the analysis, the task is set for autonomous implementation, and its executor is determined. The team can appoint a developer from any of the groups identified above as the executor. It guides by the staffing situation of the project and other considerations in

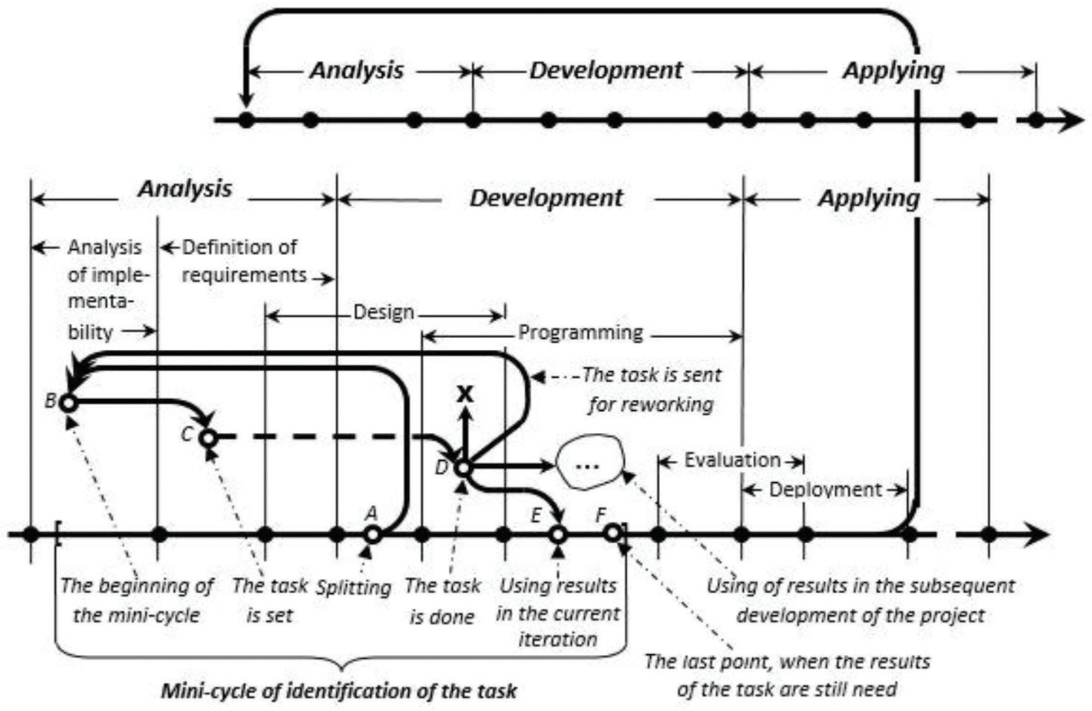
Analysis
Development
Applying
Analysis
Development
Applying
Analysis of imple-menta-bility
Definition of requirements
Design
Programming
The task is sent for reworking
B
C
D
X
...
Evaluation
Deployment
A
E
F
The beginning of the mini-cycle
The task is set
Splitting
The task is done
Using results in the current iteration
Using of results in the subsequent development of the project
The last point, when the results of the task are still need
Mini-cycle of identification of the task

Figure 2. Splitting of the life cycle.

this appointment. In particular, the team can choose an optional developer involved. We are now discussing this case.

- Analysis and statement of the task end at some point in the lifetime line, designated as ***C***. This is the moment when the optional developer involved started working. His activity, shown by a dashed line, ends at the decision point ***D***.
- Analysis of the results obtained in the point ***D*** reduces to the verification of products with the task and, if it is accepted, the results may be included in the project. This activity may lead to the following options.
    - The required results do not match the expected from the optional developer's involved work. The diagram indicates this as **X**. Key developers in this situation must make a decision regarding the executor of the assignment. In particular, is it worth continuing cooperation with this optional developer involved.
    - The work done needs to be improved. The diagram indicates this as the arrow leading to point ***B***. Key developers in this situation must repeat analysis to set updated task and, possibly, to provide assistance to the performer.
    - The results obtained can be used in the further development of the project beyond the current iteration. The diagram indicates this as cloud with "…" text. This situation requires a special evaluation of the developer's activity, on the one hand, and on the other hand, work on drawing up specifications for the product to use. At the same time, key developers should determine whether the interaction with the optional developer involved was productive and comfortable and give recommendations on how to continue cooperation in the future.
    - The results obtained satisfy the requirements, which are necessary for their use in the current iteration both in quality and in terms of execution time. The activity of key developers is the same as in the previous case.

From the suggested discussion of the mini-cycle of implementation of the task by the optional developer involved follows that the analytical work ceases to be a separate stage in the project development life cycle. It becomes a constantly performed production function of the project, the content of which at the beginning of the cycle is the formulation of tasks. When the task is completed, the content of this function (at the evaluation stage) is to prove the feasibility and appropriateness of using the results of the task in the project.

4. Personnel coordinator

The roles of executors of the project for the developed methodology remain the same as in the traditional schemes. The concretization of roles depends on the methodological scheme adopted by key developers. The same applies to the problem of compatibility of roles. The only change is the distribution of roles between performers, which should include third-party developers. Consequently, it is necessary to solve the task of staffing the project, considering it as a special *production function* that is constantly implemented during the development of the project. It is associated with the special role of the *personnel coordinator*. Usual methodologies do not consider this role specifically, because believe that the project manager is engaged in the team formation at the initial stages of the work.

The role of the personnel coordinator is not reduced to the functions of a human resources officer—a personnel department employee, usual for the traditional style of doing business in the organization in Russia. It does not come down to the work of a more habitual HR manager (Human Recourses Manager). The main difference of the new role is that it assumes the real project work (predominantly design) of the personnel coordinator, while the human resources officer deals only with technical issues of hiring and firing, and the HR manager in addition to them—training of personnel with soft skills and organization of necessary enterprise developers' attraction. The personnel coordinator performs only the last function of the HR manager, although with a slightly different content. We must emphasize that the HR manager plays the role of the administrative structure of the enterprise, which, although it affects the project activity, but indirectly, creates conditions for the work of the project teams by allocating resources to them. If we talk about unstable teams, for the most part, they are almost independent of any kind of administrative superstructure. Most often, such teams only rent premises and pay for the use of infrastructure on contractual terms. They do not form an administrative structure with a special personnel department as a rule.

The new functions of the personnel coordinator are as follows:

- Receiving and processing a task that the team offers to constantly working developer or optional developer involved.
- If a project team tasks an assignment to a constantly working developer, the interaction of the personnel coordinator with the development team in the course of the assignment is subjected to the methodological scheme adopted for the project.
- Formulation the task for an optional developer involved as a proposal, independent of the project.
- Finding a candidate for the position of an optional developers involved (using all the usual methods and sources of information on potential candidates).
- Monitoring the activities of the optional developer involved associated with the task.
- Evaluation of the activities of the optional developer involved in terms of the desirability of further cooperation with him/her.

Depending on the scope of the assignment task (scope of work planned, time constraints, etc.), the personnel coordinator becomes either a project manager formed for an autonomous task or a responsible implementer of the task. Autonomy of a task is limited to planned control activities that are consistent with the main project.

The role of a personnel coordinator can be assigned to a manager or a dedicated employee, but more efficiently when it is distributed among key developers. The advantages associated with such a distribution are a wider coverage of candidates, which means that it is possible to attract employees on a competitive basis.

5. Basic provisions of the methodology

As already noted, one of the problems of an unstable team is the lack of a suitable project development methodology. However, the forceful implementation of the methodology is unacceptable for solving the problem, since any of them presupposes the consent of the developers and mindfulness, which is impeded by instability. Therefore, in an unstable team, the methodological scheme of the project activity, as a rule, develops spontaneously and with violation of many generally accepted canons, but taking into account the human factor. This is what Cockburn calls for in his noteworthy work [4].

If we exclude from consideration methodological aspects related to interaction with the project customer, who often define them for the unstable team, then taking into account the specifics of the project activities of the team, we can recommend Highsmith's approach ASD (Adaptive Software Development) [10]. ASD is not a methodology in the strict sense of the word, but a kind of a template. Following principles of this template, the developers can choose a methodology, adapting it to the project and to the characteristics of the team. The approach focuses on teams in which training and cooperation as fundamentally important working conditions are considered. Highsmith calls for a flexible attitude toward planning: "in an environment that requires adaptability, planning is a paradox" and believes that deviations from the plan lead to objectively determined decisions. Consequently, developers should consider them not as errors or wrong attitude of performers to work, but natural, "correct" and needing to study circumstances that are the true causes of problems. Project management should aim at providing communication, as developers themselves are able to find answers to emerging issues.

From the just presented review of the ASD concepts, it is clear that they correspond to situations usual for unstable teams. The concrete definition of the methodology for project management for them that meets the principles of ASD can be any; however, the operational steps that involve a methodology for project management activities should be coordinated with the basic scheme of the adaptive development that is the reason of all methodologies that adopt the concepts of ASD (see **Figure 3**).

As one can see from **Figure 3**, in the developing project, there are three overlapping phases that are distinguished: *consideration* → *cooperation* → *learning*. They cover the entire development of the project, divided into stages, represented by rectangular blocks. The evaluation works allocate to the stages, the content of which is associated with the quality result reviews and with the processes of their obtaining. The results of the competitive works (blocks L1, L2, and L3) are evaluated to determine decision making. If satisfactory results are not obtained, and if new tasks are to be performed, a return to adaptive cycle planning occurs, the next iteration of which takes into account the process quality assessment. This cycle plays a dual role as follows:

- Adaptation to the emerging conditions of project management on the basis of the experience gained.
- Learning (in different senses) in order to improve the process and its results.

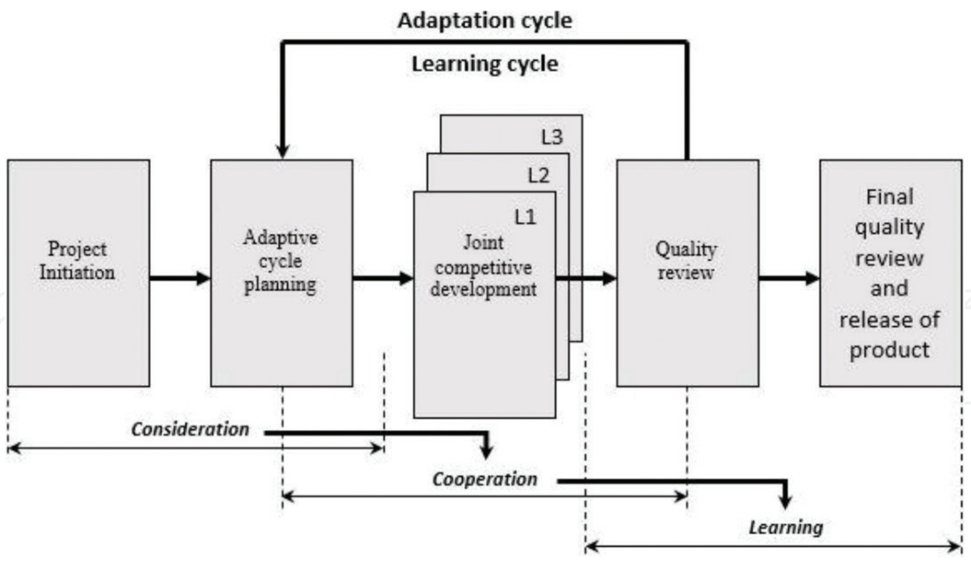

Figure 3. The development cycle of the ASD project.

Adoption of the ASD scheme for an unstable team, as a rule, does not cause difficulties even without a special discussion. As for concretization of the methodology adopted for the team, this requires the unanimous approval of key employees. The consent of the project participant with the adopted methodology of the project activity is a prerequisite for determining she/he as a key developer. The consent of the all key developers with certain methodology is a good chance of successful adopting it for the team. Realizing this, we give only a sketch of the concretization of the methodology that corresponds the ASD scheme with the usual phases of the life cycle of the project development. In this coordination, we take into account the need of splitting and organizing mini-cycles of assignments by optional developers involved, as well as the work of the personnel coordinator.

The subsequent sketch uses the concepts presented in the discussion of the life cycle of the project and the organization of mini-cycles for outsource performing tasks.

Consideration phase:

- *Initiation of the project*. This is the formation of a development task. The task is based on an external order or from the realization of the need to implement the development.

  It corresponds to the analysis stage and the beginning of the definition of requirements. The stage is associated with the initial definition of tasks that one can formalize as offers for autonomous execution.

  The personnel coordinator evaluates the resources available and necessary for the implementation of the project.

- *Adaptive cycle planning*. This is a preparation of architectural solution. For the first iteration, when an architectural decision has not yet been made, information is collected for its development. For subsequent iterations, questions are resolved for the integration of new components into the existing architecture, as well as the modernization of architecture. An important part of adaptive planning is the definition or adjustment of the quality criterion of architectural solutions.

The staffing task of the stage is the appointment of the performer to whom the team assigns the following work: the search for a ready similar component or a statement of the need to develop a new module. Priority in providing work to executors is given to key and constantly working developers. For assignments, which is decided to offer to constantly working developers and optional developers involved, a *curator* from among key developers is determined. The role of the curator is compatible with the role of the personnel coordinator. The execution of external assignments is in the responsibility of the curator.

Cooperation phase:

- *Joint competitive development of opportunities*. This is the completion of the architectural design phase and the execution of the programming phase. The stage involves the following works:
    - developing solutions for each of the tasks, if the need for this is recognized;
    - identification and implementation of works on the adaptation of selected and developed solutions to the requirements of the system.

  The execution of works by key developers is not specifically regulated. The curator of external works consults (in different forms) their performers (option developers involved) and reports to the personnel coordinator. She/he also identifies situations when the implementation of the task requires the mastering of special educational material. In this case, one of the key developers organizes the necessary classes for interested performers (e.g., in the form of teleconferences).

- *Primary quality estimation*. This is the completion of the execution of external work and discussion of the solutions obtained (overlapping works with the next phase). If one cannot choose the solutions suitable for the development of the system being developed, then a mini-cycle of learning is organized or a transition to the iteration of the adaptation cycle is done. In the discussion, the performer (key or a constantly working developer) and the curator participate. The results of the discussion are forwarded to the personnel coordinator for analysis.

  The stage corresponds to an evaluation, as a life cycle stage in that part of it that is associated with a local estimate of external results by a curator and a personnel coordinator.

Learning phase:

- *Quality review*. This is an assessment of the work carried out and the decision to continue or complete the project. The purpose of the review is to decide whether the release is ready or need to organize a transition to the iteration of the adaptation cycle.

Communications of this stage do not differ from the discussion at the stage of primary quality assessment with the only difference that all key developers are involved in them.

The stage corresponds to the completion of the "Evaluation" phase of the life cycle, during which a conclusion is made about the attestation of the results obtained, that is, about the expediency of the transition to the offer for release.

- *Final quality review and release transfer to the customer*. The content of the activity of the stage consists of determining the level of success of the work in general and the organization of activities that ensure:
    - integration of the results obtained as products for users;
    - evaluation of product quality from the point of view of users;
    - organization of works on support and maintenance (new design tasks).

  Communication of learning phase is a discussion of the success of the implementation of all three phases of the project with the aim of adjusting the future activities of the team and evaluation criteria, taking into account the experience gained, as well as interaction with users and with the customer, regulated by them.

6. Illustration of the proposed approach

In this section, we provide information about the company NetGon, whose activities served as the basis for the formation of the proposed approach to the organization of management of software projects in the conditions of personnel instability. This company specializes in the development of web applications and support tools. Information for the company's work directions, as well as on implemented projects can be found on the company's website http://netgon.ru/.

NetGon emerged as a typical IT start-up. Its creator had the idea to implement a system that could support the rapid development of small applications based on pre-prepared frameworks, templates, and technological toolkit. He managed to find a group of three like-minded people who became key developers of the gradually formed team. Currently, the company employs 15 constantly working developers and approximately 30 optional developers involved. Key developers keep a record of specialists involved in one-off works. Creator NetGon is the manager of the company.

The methodology of project activities in the company evolved spontaneously and based on the experience of previous work. This is usually for beginners. The team realized the need for an appropriate methodology when there was a feeling of a loss of control. In response to this, the team introduces the rules for working with the repository of the basic components. Then, it began to use the *evaluation cards of the employees* involved. These cards maintain information about the strengths and weaknesses of employees, their competence, prospects, and so on. Key developers consider and use the permanently updated database of such cards as a tool to mitigate the staff instability of the team. Optional developers involved, who performed one-time tasks well, receive the status of constantly working developers, and the status of those who did not quite cope with the tasks is reduced perhaps to the level of an undesirable candidate.

At the same time, key employees analyzed different project management methodologies and their tools. As a result, key developers identified the approach of the ASD, which turned out to be conceptually close to the style of work of the company. The provisions of this approach, associated with the development of their own methodology for project implementation, attitude to errors, and the requirement of adaptability to rapidly changing working conditions have become the basis for improving the implementation processes of both custom and initiative projects.

On this basis, the company distributes the work among the developers, monitors the performance of tasks fulfillment, controls the quality of work results, and builds relationships between employees. Very attractive especially for hired employees are the so-called training sessions, on which the developers share their experience in implementing projects, inform the participants of impressions about the materials they read, and so on. These regular and episodic classes use remote communication tools.

As noted earlier, with respect to optional developers involved, the listed activities are the content of the production function that determines the role of the personnel coordinator. This role is shared between key employees who are assigned to track optional developers who are involved in work related to the tasks assigned to them. Designated employees, not optional developers, are responsible for the task execution process. They represent this process for the team.

These and many other organizational and technical measures allowed the company to improve its efficiency significantly. For instance, if in the initial period of the company's work the time spent on creating the project was from a week to a month, then after the adaptation of the ASD approach, it stabilized for most projects within a week. Initially, the company consisted only of key and optional developers. The group of constantly working developers was gradually formed. It grew by increasing the number of optional developers involved and identifying among them the most reliable and interested employees. This group stabilized when the company began to organize training sessions.

It is worth noting that the NetGon team has a need to disseminate the accumulated knowledge and experience gained in real projects. This is a direct consequence of the orientation toward the ASD approach to the organization of project activities, which involves the learning of employees. As a result, the company offers for the market a number of special products, the content of which is directed to training on web-programming techniques. Today, it is too early to talk about the real demand for these products. Nevertheless, the initiative itself deserves attention.

7. Methods of increasing maturity and reliability of an unstable team

The illustration just presented demonstrates the development of the maturity of the company, which sought to achieve stability through the using of a suitable methodology and organizational solutions. The analysis of this experience makes it possible to present the method of overcoming instability problems that was obtained as result of the company development. It is based on using some indicators important for effective evaluation of the company:

- Percentage of unsuccessful projects.
- The share of complex projects in the company's works.
- The stability indicator of a group of constantly working developers.
- Conformity of constantly working developers to the qualification requirements of the execute and perspective projects.
- Growth or decrease in the requirement to use optional developers involved.

We do not consider here indicators related to the economic evaluation of the company: profit, profitability, costs, and so on. For the choice of methods of strategic planning and evaluation of the solutions to be adopted, we recommend the reader to refer to the special literature (see, e.g., [11]). At the same time, we emphasize the importance of using economic indicators for monitoring the state of processes in the company and in the case of unstable teams.

How selected indicators are measured depends on the characteristics of the company and its projects. However, to increase the company's maturity and, in particular, to overcome the problems associated with instability, it is necessary for key employees to agree on how to use the adopted indicators to evaluate the company's performance.

Ignoring these problems leads to the degradation of the company and, ultimately, its destruction. Most of the programming teams whose activities we observed sooner or later were forced to go through the process of reorganizing the spontaneous processes and reach the maturity level, which in the CMM classification is called repeatable. Unfortunately, some of them lost their independence and in fact became an external unit of the large companies, forgetting about their initial plans and intentions. In this respect, NetGon is one of the few good exceptions.

For NetGon company, the first indicator changed significantly. If during the first year of the company's existence, there were about 30% of unsuccessful projects on average, by the end of the year, the percentage of such projects stabilized at 10%. The second indicator has grown. The share of projects that last more than 2 months has increased to 30%, whereas before the reorganization of the development processes, the company could not afford more than two such projects (one of them is an initiative project). The third non-quantitative indicator is very important for assessing the quality of the company. It can be estimated by the number of stop-working employees from a group of constantly working developers, but taking into account their importance for the company's activities. There were no such employees in NetGon. To determine the fourth indicator, key employees should take into account not only the current needs but also the development strategy adopted for the company. Using this information, they assess the need for the development of the company's personnel and, thus, establish qualitative and quantitative parameters for the process of attracting additional employees. This information serves as a guide for the fifth indicator: deviation from the specified parameters is undesirable. Today, in NetGon, the only task of attracting external developers is to test them as candidates for the constantly working developers. This is considered as an element of the company's personnel policy.

We pay considerable attention to the discussion of maturity indicators, because the use of them by key developers allows *to identify and understand the problems of the team that, in particular, hamper the involvement of the good developers*. In its essence, this is the first step toward increasing the maturity and reliability of an unstable team.

The second step toward increasing the company's maturity is *to identify the risks that arise from unsatisfactory indicators and to collect proposals for mitigating them*.

The work of the team with the optional developers involved is always associated with certain risks for the project. First of all, these are unjustified expectations of performing the tasks with due quality and in a timely manner. *Mitigating of this risk is usually attained with entering into contracts with third-party developers* and, if agreements are reached and formalized legally correctly, this can guarantee against financial losses.

Unfortunately, the contract cannot provide for threats related to inaccurately or incompletely formulated requirements for the program project and, possibly, to the tasks proposed for outsourcing. In such cases, the company has a natural need for a suitable methodology for the project activity. *The introduction of such a methodology is one of the important methods for increasing the company's maturity*.

Adequate project methodology only contributes to mitigating risks, but does not exclude them. Interacting with constantly working and optional developers, the project manager can and should apply other *well-known techniques for risks mitigation.*[1]

Permanent contact with employees and prompt correction of decisions are one of the conditions of the risk mitigation policy. When solving tasks of optional developers involved, the opportunities for the contacts with them are significantly limited. Therefore, *the tasks proposed for outsourcing should be set so that the expected results are useful, regardless of the possible changes in the projects that are being implemented*.

The teams that we were able to observe often made mistakes due to the failure to fulfill this condition. Some of them had to abandon the use of the optional developers involved. And this can be considered as a radical method for solving instability problems. However, those of them that correctly evaluated failed were able to determine what tasks can and cannot be outsourced. As a result, they got quite good opportunities for effective using of outsourcing. For example, NetGon today trusts up to 60% of project works for the optional developers involved and, accordingly, reduce the costs of development. Thus, *the thoughtful selection*

[1]This topic is beyond the scope of this chapter; for its study, we recommend the publication of [12] on the MITER resource (https://www.mitre.org/) and other sources to which it refers.

of tasks for the optional developers involved mitigates the risks of inefficient use of outsourcing by unstable teams.

This statement is necessary, but not a sufficient condition for the success of using outsourcing. If the team does not care about managing the implementation of external tasks, the effectiveness of third-party developers can significantly decrease. That is why we single out a special production function for the project, whose purpose is to monitor the performance of external works. This function is related to the role of the *personnel coordinator, whose definition we consider as the base of the proposed approach to increasing the maturity and reliability of the unstable team*.

The role of the personnel coordinator can be distributed between key or constantly working developers, but for each optional developer involved or for a group of such developers who are jointly performing a task, there should be a single supervisory coordinator. The area of responsibility of the coordinator includes communications with optional developers involved and the management of outsourced tasks. It represents to the team both the results and the progress of the tasks. If necessary, the coordinator organizes staff training for both optional developers involved and, possibly, other employees.

The successful development of any company depends not only on the maturity of the projects development processes but also on the attractiveness to potential developers. An important component of success is the formation of an inviting reputation: caring for employees, creating comfortable working conditions, providing opportunities for professional development, career prospects, and so on. This is very important for a company that, in conditions of instability, seeks to increase its maturity. Although the issues of reputation enhancement are beyond the scope of this chapter, we note that the ASD methodology, which we discussed earlier, is very productive in conditions of instability. *Three phases of the project: considering, cooperative, and learning, which involve all developers, create conditions for the active joining external employees with the team members*.

8. Conclusion

In this chapter, we presented methodological proposals for organizing the activities of programming teams that attract external performers for carrying out projects. A list of typical works for which such outsourcing can be useful is described as follows:

- Implementing a new or adapting an existing toolkit with precisely defined functions required for the execution of a project.
- Development of library facilities that meet the fixed conditions for their use.
- Optimization of the developed components, and/or adaptation to special equipment.
- Consulting services of various kinds: clarification of user requirements, analysis of the target market for the project products, recommendations on the instruments used, and so on.
- Audit of project and organizational activities of the project team.

Teams that are forced to use outsourcing, we conditionally call as unstable, to emphasize obvious for them staff variability. The successful experience of external outsourcing by teams that consciously focus on the main directions of the project and pass on to others what is not fundamental deserves attention. Observations and analysis of the experience of several such teams allowed us to identify general principles, the adherence to which we can recommend in similar situations.

The approach to the specification of the methodology in order to achieve compliance with its accepted conceptual scheme is presented in the alignment of the classical stages of the life cycle with the scheme of adaptive development of the ASD project. One can consider it without relation with the particularity of unstable teams. When adapting any methodology to specific conditions, it is necessary to solve a similar problem. It is especially important to raise this problem if the methodology of teamwork is spontaneous, and the team needs to systematize the processes of projects development and use a methodology that corresponds to the working conditions of the team. At the same time, we should not absolutize the proposed constructions. The concretization of the methodology depends on so many factors, and therefore it would be naive to believe that it is possible to give “universal” recipes for solving this problem. The only thing that we can here recommend is the awareness of the choice of organizational forms of project activities based on the study and critical analysis of existing approaches.

Acknowledgements

This chapter presents the work that is supported by the grant RF RSF №. 14 11 00485 “High-performance methods and technologies of electrophysical processes and devices modeling.”

Author details

Igor Nikolayevich Skopin[1,2]*

*Address all correspondence to: iskopin@gmail.com

1 Institute of Computational Mathematics and Mathematical Geophysics of the Siberian Branch of the RAS, Novosibirsk, Russia

2 Novosibirsk State University, Novosibirsk, Russia

References

[1] Futrell RT, Shafer DF, Shafer LI. Quality Software Project Management. Prentice Hall PTR; 2002: 1677 p. ISBN: 0-13-091297-2

[2] Fowler M. The new methodology. 13 December 2005. https://www.martinfowler.com/articles/newMethodology.html [Accessed: 2017.07.25]

[3] Brooks Jr. FP. The Mythical Man-Month: Essays on Software Engineering / Anniversary ed. Includes bibliographical references and index. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.; 1995. p. 322. ISBN: 0-201-83595-9

[4] Cockburn AR. Characterizing people as non-linear, first-order components in software development—Humans and Technology. HaT Technical Report [Internet]. 1999. http://alistair.cockburn.us/Characterizing+people+as+non-linear,+first-order+components+in+software+development [Accessed: 2017.07.25]

[5] DeMarco T, Lister T. Peopleware: Productive projects and teams. 3rd ed. New York: Dorset House; 2016

[6] Microsoft Visual Studio. Scrum 1.0 [Internet]. Available from: http://visualstudiogallery.msdn.microsoft.com/59ac03e3-df99-4776-be39-1917cbfc5d8e/ [Accessed: 2017.07.25]

[7] Paulk MC, Curtis B, Chrissis MB, Weber CV. Capability maturity model, version 1.1. IEEE Software. July 1993:18-27

[8] Skopin IN. Training in the remote development team in a series of projects for web systems. In: Sci. Works of The 2nd All-Russian Scientific-practical Conference "Information Technologies in the Education of the XXI century". Vol. 2. 2012; Moscow. Moscow: National Research Nuclear University; 2012. pp. 279-282 (in Russian)

[9] Skopin IN. Fundamentals of Software Project Management. Moscow: Internet University of Information Technologies; 2004. 363 p (in Russian). ISBN: 5-9556-0013-2

[10] Highsmith JA. Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. New York: Dorset House; 2000. 392 p. ISBN: 0-932633-40-4

[11] O'Sullivan A, Sheffrin SM. Economics: Principles in Action. Upper Saddle River, New Jersey: Pearson Prentice Hall; 2003. p. 550. ISBN: 0-13-063085-3

[12] Risk Mitigation Planning, Implementation, and Progress Monitoring. In: MITRE Systems Engineering Guide. https://www.mitre.org/publications/systems-engineering-guide/acquisition-systems-engineering/risk-management/risk-mitigation-planning-implementation-and-progress-monitoring [Accessed: 2017.12.27]