Building a Hybrid Process Model for a Complex Software System Integration

K. Smojver, H. Belani and Ž. Car
University of Zagreb, Faculty of Electrical Engineering and Computing,
Department of Telecommunications, Zagreb, Croatia
{karlo.smojver; hrvoje.belani; zeljka.car}@fer.hr

Abstract—In order to fulfill every project-specific requirement in software development today, and produce a software product with desired functionality and satisfactory quality attributes, adequate development and management processes ought to be designed, implemented, adopted, operated and/or improved. Mandatory requests on shorter time to market, staying within the budget limit and producing high quality software become even more stressful for project stakeholders when dealing with complex software systems, like the ones for enterprise resource planning or multidimensional drafting of complex models for technical and engineering products. Based on the real-life experience, that is thoroughly analyzed here, in integrating such two complex software systems, a hybrid process model for project management and software development in agile environment is proposed in this paper. The model attributes specially stressed out and analyzed are suitability for agile conditions and adaptive behavior. Conclusion gives usage perspective for such a hybrid model, with limitations discussed and future work proposed.

Keywords— project management, system integration, complex software, agile development, process modeling

I. INTRODUCTION

Intensive software development for almost all known purposes of human engagement is an evident fact in today’s world. Software is indispensable in accomplishing different business services, e.g. banking, healthcare, space research, etc. In order to fulfill every project-specific requirement and produce software with satisfactory quality attributes, adequate software development, but also project management processes ought to be designed, implemented, adopted, operated and/or improved. It is not nearly trivial to make all project stakeholders follow sometimes very strict guidelines, and process activities that maybe don’t easily map to given tasks. The stress is significant while trying to comply with requests on shorter time to market, staying within the defined budget and producing high quality solutions [1].

When preparing a project there are several organizational issues, which must be covered in order to reach positive results at the end of the project execution. A software development project has to be carefully planned before it is embarked on. At first, it is very important to define a process model. One has to decide upon the exact process model for each project to be followed: which activities are being undertaken, which milestones are to be identified, how to ascertain whether these milestones are reached, and which are the critical paths. The next step is the organization of the project.

The project plan has to state which information, services, resources and facilities are to be provided by the users and when they are to be provided. Within the project team, various roles should be identified: project manager, tester, programmer, analyst, etc. One has to clearly delineate these roles and identify the responsibilities of each of them. Further step is defining the standards, guidelines and procedures that must be followed by all project members in a software development project [2]. Hybrid process model proposed in this paper provides a complete set of concepts to deal with complex software integration issues. The goal of the proposed model is to create transparency without heavyweight processes suitable especially for novice developers [3].

The structure of this paper is as follows: section 2 describes drivers for and challenges of complex software system integration, suited into today’s agile business environment, while section 3 gives thorough analysis of the corresponding case study conducted as a real-life project, and elaborates issues of project execution, communication and risks, quality issues and results visibility and documentations. Section 4 gives a brief overview of two agile project management and software development methodologies, which served as a ground for a hybrid process model proposal, which is described in details in section 5. Proposed model evaluation and comments on related work are given in section 6. Conclusions are given in section 7, with suggestions for future work.

II. COMPLEX SYSTEMS INTEGRATION

Integration of complex software systems is a difficult task, and it is not easy to ensure the quality of developed software. Integration directs on developing large, complex systems in a manner that facilitates robustness of products, economy of the development process and rapid time to market. Integration focuses on the construction of more abstract architectural elements supported by a process model. Besides representing a plan for an actual integration of the complex software system, another key purpose of the model is to allow reasoning about important aspects of software projects before actually performing it.

Project management methodologies are broader and cover fundamental business processes, such as: managing costs, human resources, procurement, etc. Software development methodologies are often an integrated part of project management methodologies because they are in charge for development of a specific software product. These methodologies are often merged into one single
methodology if it is suitable for the project and its surroundings [4]. Many modern companies worldwide use complex software systems in order to provide high-quality support to their business processes, e.g. concerning development of large technical design models, or management of products and resources. Complex systems connote enterprise resource planning (ERP) software such as SAP, Navision, Epicor, PeopleSoft, etc. Other segments of complex systems are specific applications for computer-aided multidimensional drafting of complex models for technical and engineering products, such as CAD (computer-aided design), CAM (computer-aided manufacturing), CAE (computer-aided engineering), etc.

When several different complex systems, e.g. ERP and CAD, are used in the same company, a justifiable business need for integrating such systems occurs; almost immanently. The ultimate goal of integrating such systems is reducing time to market and easing up the product lifecycle management [5], as shown in Figure 1. Such integration would offer a possibility to predict possible dates and expenses of research and development efforts, production itself, the product finishing and delivery during the design phase.

Basic functional requirements for the integration are transferring any kind of drawings between CAD and business support systems. Data transfer must be fast, bidirectional, reliable and secure. One of the most important requirements is data integrity. Since many people could work on the same model at the same time, data redundancy or even loss of data is possible. So, it is mandatory to ensure safe and bulletproof data management with clear distinction who can create, read, update and delete (CRUD) sensitive data, when and under what circumstances.

These requirements are performance-related but seem obligate, while other functionality is optional. On the other hand, the daily usage of the integration functionality must be as simple as possible.

III. THE CASE STUDY EXPERIENCES

In this case study a project of optimizing complex systems integration is described. A word optimizing regards speeding up an existing data transfer between ERP and CAD systems.

A. Project Characteristics and Initial Plans

Project was characterized with a complex existing integration, which required a wide programming knowledge. Program code was poorly documented and incoherentely written because of constant pressure during development of previous version of integration. Members of project team were physically distributed, with different language and cultural manners. Agreed dates and deadlines were setted, so first results had to come up soon after the project beginning. Expected project results were concentrated in the domain of enhancing data transfer speed, and the functionality had to be kept on the same level. Generally speaking, there were no specific plans on how to realize expected project results, one and only plan was to achieve the ultimate goal – speeding up existing integration.

B. Project Execution

Execution of the project was much different than it was initially planned, although initial plans were quite superficial. Project execution will be described through a few standard project features like roles, quality, risks, documentation, etc.

1) Planning and Roles

Planning is one of the most important project processes and it should not be neglected in any way [6]. This project was characterized with rather superficial planning activities. All of the plans were focused on the final goal, and not on the executing project work. The biggest problem of project team was a weak correlation between individual and collective plans. This means that personal plans of team members were not synchronized, and it caused many misunderstandings. Team member’s roles were changed a few times during project execution.

A. Integration Architecture

Important feature of such complex systems integration, regarding the software architecture, is that it must be invisible for end user, in a way that it must not require additional actions from the user during his work. It is desirable for the user to see only the results of data transfer between CAD and business support system, like it is shown in Figure 2. The physical implementation or architecture is not important in the first place, as long as it offers desired functionality.

Today trend in integration of complex software systems is so called multi-stage-integration which is technique for slicing software system integrations into smaller and more manageable pieces. This allows the easy detection of successful or not-successful integration paths. By combining different data sources available within a development environment, a high degree of transparency is achieved.

B. Integration Performance

Concerning the complex system integration performance, it must support distributed working environments for large project teams, and provide real-time communication with connected systems (ERP and CAD). It must be scalable and flexible, in order to manage a vast number of client requests.
After the project started, roles disposition had to be changed because it became obvious that some team members were not competent enough for completion of assigned roles. It is also important to stress out that each team member had few roles, e.g. programmer, tester, designer, etc. Role multiplicity is generally positive project characteristic, but it is undesirable under conditions of weak planning and organizing activities. Interesting fact about this project is that the role of a project manager (or leader) didn’t exist, but during the project execution it was evident that this was a serious drawback.

2) Communication

A good communication was hard to realize because the project team was distributed. So, there was no everyday eye-to-eye contact between team members. The basis of communication between team members were telephone, e-mail and applications for remote desktop connection. There were a few team meetings and the main purpose of these meetings was getting to know with all team members. Communication was not permanent, and was rather unorganized. Communication with stakeholders was stable and regular, but the communication between team members was superficial. Another problem of communication in this project is caused by staged communication, so the initial meaning of sent information was changed at the end. Groupware software could improve communication between members of distributed project team [7], but there was no useful time for implementing one during the project.

3) Risks and Issues

A good risk and issue management is exceptionally important in any project, and can affect project success significantly. This project risks were not estimated thoroughly, so there was no preventing or fixing plans. The most hazardous risk was overstepping time deadlines. The other important risk was team member’s incompetence, which could cause project failure [8]. Unfortunately, this risk was actualized, so competent project staff had to be engaged. Another unpredicted risk was connected with wrong estimation of time needed for new team members to understand the principle of work of existing integration. A negative consequence of that was a need for engaging staff which has been working in project of developing previous integration. There were also many issues which have been prowling this project. One interesting issue was change in the proprietary structure of product customer.

C. Quality and Metrics

Quality represents closeness of project results and client’s demands [9]. In this project quality of a new software product was controlled via test cases and progress presentations. A few times during project execution client asked to see progress of project work. Other aspects of quality were not in focus. Metrics is closely connected with quality and it represents improvement or deterioration of results quality [9]. Metrics management is one of the most complex project management processes, and usually it is used in big projects. In this project there was no person in charge for managing metrics, so these processes were not performed at all.

D. Documentation

Documentation is extremely important part of each project of software development [10]. Importance of documentation can’t be stressed out enough - it is the only permanent way of storing company knowledge. Documentation of previous version of integration was poor and insufficient for getting a thorough insight of integration work principle. During this project there was no process or systematic approach of creating project or technical documentation. Project documentation, like team experiences or ways of solving some problems, was completely neglected, whereas technical documentation has gotten more attention. User instructions were written in a proper way because it was one of the client’s primary requirements. Other documentation which is aimed for better understanding of program code and principle of work, like UML diagrams and code annotations, has been written superficially and in inadequate way.

E. Project Results

Project results can be partitioned to visible and invisible results. Visible results are the ones which end user can perceive during the use of product, e.g. data transfer speed, functionality, whereas he cannot feel invisible results, e.g. program code overhead and inconsistency, insufficient development documentation, etc. In this project visible results were exceptionally good, especially in the context of increasing data transfer speed. On the other hand, invisible results were not satisfying, mainly because development documentation was poor. Generally speaking, this project was successful mainly because the clients were satisfied with results, but all the future projects, which will work with this integration, risk to experience difficulties with the understanding of basic working principles of this integration.

F. Methodologies

From the previous project description it can be recognized that project execution was rather unorganized and some results were not satisfying. One possible way of fixing these mistakes and improving processes and results in following projects are to use adequate project management and software development methodologies. Use of such methodologies strongly encourages planning and better organization of project execution.

The correlation between project management and software development methodologies is imminent. Project management methodologies are broader and cover fundamental business processes such as: managing costs, human resources, procurement, etc. Software development methodologies are often an integrated part of project management methodologies because they are in charge for development of a specific product [11].

![Diagram](image-url)

Figure 3. Conceptual relation between project management and software development methodologies
These methodologies are often merged into one single methodology, as suggested in Figure 3, if it is suitable for a project and its surroundings. Some of this project’s characteristics were:

- Small project (about 5 months, 5-6 people),
- Geographically distributed project team,
- Possible changes of requirements during project execution (requires adaptive approach),
- Object-oriented design approach,
- Simple and manageable methodology,
- Financially less demanding solution,
- Not too radical methodology – inexperienced team members.

IV. REFERENT METHODOLOGIES

According to previously given characteristics, a new hybrid process model is proposed out of several components of referent project management framework and software development methodology [12].

A. Referent Project Management Framework

Main requirements for project management framework are agility and adaptive behavior. Reason for this lies in the fact that there were possible changes of requirements during project execution and short delivery time [13]. Some of the necessary fundamental principles of referent framework or methodology are:

- **Client-orientation** - Project processes should be managed in collaboration with client. The client has ordered a product, so the product must be tailored according to his aspirations;
- **Fast and frequent results** - Small and frequent project deliverables are desirable because they can be easily presented to customer. That way the client is informed about the project progress continuously and the above mentioned fundamental principal is fulfilled;
- **Continuous research** - At no point during the project, team members should think that their research is finished. There are always better solutions which can be adopted, of course, if they do not change project plan drastically;
- **Change is welcome** - In small projects customer change requests are normal, and should be accepted if it improves the current idea, within agreed project deadlines and budget;
- **Not necessary complex long term plans** - Detailed long term plans are good when it is possible to work in intact project conditions. But, since changes and turbulent conditions are assumed all the time, it is almost impossible to create such plans.

For given conditions the most suitable is ICONIX software development methodology. ICONIX can be placed into the middle, between RUP, Extreme Programming (XP) and Agile Software Development [16]. It takes the best of above mentioned methodologies and minimizes development process. ICONIX is based on UML diagrams like RUP, that’s why it is so suitable for object-oriented development. ICONIX also supports iterative development and copes well with rapid changes of scope, design, requirements, and estimations [17]. In ICONIX phases of analysis and design are exceptionally important and strongly bounded. ICONIX methodology is chosen since it combines in the best way agility needs and required level of process formality.

For given conditions the most suitable is ICONIX software development methodology. ICONIX can be placed into the middle, between RUP, Extreme Programming (XP) and Agile Software Development [16]. It takes the best of above mentioned methodologies and minimizes development process. ICONIX is based on UML diagrams like RUP, that’s why it is so suitable for object-oriented development. ICONIX also supports iterative development and copes well with rapid changes of scope, design, requirements, and estimations [17]. In ICONIX phases of analysis and design are exceptionally important and strongly bounded. ICONIX methodology is chosen since it combines in the best way agility needs and required level of process formality.

V. HYBRID PROCESS MODEL PROPOSAL

Proposed process model shown in Figure 4 is composed of components taken from referent framework APF, ICONIX methodology and some general project processes. More general processes are taken from APF, and development specific processes from ICONIX. White boxes in Figure 4 represent core activities of the model; grey boxes represent groups of activities (phases) according to APF; curly brackets represent general sub-
process groups which are common to many projects and they are not as specific as APF phases.

The most important part of this model is black arrow which stretches between activities of code and model revision and technical requirements for iteration. It represents the end of current cycle and the beginning of new iteration. That is what stresses out agility and adaptive behavior of this process model.

Proposed model consist of four main activity groups. In each activity group there are several activities which are composed of specific techniques, tasks and artifacts. Project preparation activity includes all the tasks which are necessary to be carried out before the start of analysis, such as: selecting project manager, gathering team members, defining project approach, etc.

Activities located in the Initial estimations activity group are focused on creating rough plans without many details. After all, it is not possible to make a detailed plan on the beginning of a project in such variable conditions (possible requirements change). In this phase it is important to gather functional requirements from customer and determine conditions of satisfaction. This information is the base for creation of initial plans, such as general WBS, "touch and go" cost, time, risk and resources estimation, etc. So, all critical or very specific estimations should be avoided in the first phase, not to preclude new ideas which could come up.

Next activity group consists of activities intended for creating detailed plans for current iteration. This phase is far more specific and harder for successful realization than the previous one, because it requires more detailed approach in every aspect (management and technical). Technical experts must analyze client’s requirements, and according to that create adequate development diagrams (UML diagrams) and software design elements. At the same time, project manager should estimate costs, required time, recourses, and risks, for current iteration. A good prediction of possible risks can be very meaningful, and could save a lot of time in case the foreseen risk actually happens. It is important to stress out that this phase is extremely important for the sake of iteration’s accomplishment and success.

After the planning phase, comes the iteration execution activity group. It consists of activities aimed for finalizing design, programming and testing. This phase is actually a “materialization” of previous planning phase. First of all, it is necessary to precisely define the behavior and interference of all development entities, and then review the complete design to verify its applicability. The next step is code development and testing on the source code level with JUnit test cases.

Final activity group represents a checkpoint, during which a client must carry out his opinion and contentment with shown results. In that activity group some smaller errors are eliminated and design models updated if necessary. In this phase the project team should ask itself a few questions, in the context of current iteration, such as: What have we done well? What can we do better? What problems do we have? What have we learned? Answers to these questions should give to team members a small review of finished iteration right before the entrance to the next iteration.

After that, project goes into a new iteration or ends up in the final phase – project closure. During the project closure activities all experiences and knowledge gathered during the project should be consolidated, in order to provide useful information for all future projects.

In this model it is noticeable it stresses out the importance of planning and design, because these two phases overlap and are inseparable. It is also interesting to notice that planning is divided into two big phases – phase of rough plans, and phase of detailed iteration planning, which is also a characteristic that emphasizes adaptive behavior.
VI. MODEL EVALUATION AND RELATED WORK

In each of above mentioned activities, some project role activities are very intensive. The activity plan of these roles can be miscellaneous and flexible, depending on the specific needs of project. Some potential roles are: project manager, programmer, software architecture designer, documentation creator, tester, end user, etc. All possible project roles can be grouped into four main role groups, which are: project managers, developers, users, and project sponsors. Their activity is not constant during the whole project, but on the contrary, it changes all the time during the project execution.

As it can be seen from the figure below, the project manager is constantly active during a typical software project, because he is main responsible person for the project desirably successful outcome. Development team starts being active in initiation phase, and after that is under full throttle all the way until the end of project. Clients are active in specific moments of the each iteration, and project sponsors are mainly active during the project beginning and the end.

![Role activity and time estimation overview for each typical software project phase](image)

Although this paper discusses some process-based approaches to complex software systems integration, there is some significant work in this field that points out the central role of software architectures for solving integration issues [20]. They showed how architectural reasoning was used to design and compare integration alternatives, analyzing four levels of integration: interoperation, enterprise application integration, integration by common data model, and a full integration. Furthermore, they propose to include risk analysis into architectural analysis, or vice versa, to make it more explicit, in order treat risk together with other quality properties [21][22]. To address these and other important concerns during integration planning and execution, it is often needed to organize agile and adaptive approach to software development and project management activities, and integrate them also. This paper contributes in this direction.

VII. CONCLUSION

This paper discussed different potential methodologies for complex software system integration, with the real-life project case study described. Experiences from the case study and lessons learned have served as a base for a new hybrid process model for complex software system integration. The proposed model can further be evaluated and utilized for other project cases, such as in-house integration projects in company mergers.

Experiences from the literature show that sometimes, in the case of the smaller projects, it is suitable to use hybrid process model based on the specific project management and software development methodology, instead of dealing with concerns and consequences of two separated methodologies.

ACKNOWLEDGMENTS

This work evaluates results from real-life software project that occurred in period of five months, lasting from November 2007 till March 2008.

This work was carried out within the research project “Content Delivery and Mobility of Users and Services in New Generation Networks”, supported by the Ministry of Science, Education and Sports of the Republic of Croatia.

REFERENCES