Visual Modeling and Software Project Management

Dmitrij V. Koznov[1] [2]
Saint-Petersburg State University, Mathematics and Mechanics Faculty

Abstract

During last decades in Software Engineering the concept of software development process was formed where analysis and design played a significant role. Visual modeling takes an important place there. However practice of the last years has shown, that these approaches are rather a philosophy, than a guideline for creation of effective software development strategies. Concerning visual modeling it appears that frequently it is visualized not what requires formalizing and wide discussion, but what is rather easy to visualize. Thus, a question on revising a place of visual modeling in software development process is still actual. In this paper we argue that this place is project management. We introduce the idea of a conceptual field of the project as a scope of its integrity. We regard as one of the main goals of project manager to keep a balance between conceptual field activities and basic project activities. We think that visual modeling can help in this.

Keywords: management, visual modeling, software process.

1. Introduction

Brooks noticed that the main problem of a large-scale software project is the complexity of supporting its conceptual integrity (Brooks, 1975). However in a real situation it is not easy to estimate the efforts which would be both effective and not excessive. One of the main goals of the project management is to achieve this balance. The balance between "theory and practice, technology and people, customer value and provider profitability, strategies and tactics" (Royce, 1998) is emphasized as a main perspective of software project management.

For a long time many researchers in software engineering believed in visual modeling. (Ross, 1977; Jacobson, 1992; UML, 2001).

The use of visual modeling in software engineering was accompanied by creating specific diagram languages as well as diagram-based techniques: structured analysis (Iordan, 1989), plenty of object-oriented methodologies finally evolved to UML[3] (UML, 2001) and RUP[4]

---

[1] Postal Address: room 2349, Bibliotechnay sq.2, Petrodvoretz, St.Peresburg, 198504, Russia.
E-mail: dim@tepkom.ru.

[2] The author would like to thank Alexander Ivanov, Konstantin Romanovsky, Alexander Marianenko for discussing the paper and clarifying its ideas. The author also would like to mention the use of Alexander Ivanov's idea about applying visual modeling to operating the software project information that can be presented in a compact graph view.

[3] Unified Modeling Language
[4] Rational Unified Process

(Kruchten, 1998). Later the diagram-based techniques developed using UML (a good survey of such techniques is presented in (Fowler, 1999)).

The main prerequisite of the approaches mentioned above was the use of analogies with drawings in civil or mechanical engineering. This idea is actively criticized now (for example, by adherents of agile methodologies (Fowler, 2001)) because it was found hard to support the integrity of this information. Approaches that suggest using the visual modeling with code-generation techniques are not widely spread because of hard special work that needs to be done for customizing these approaches to a concrete project (Koznov, 2000). We think that code-generation techniques may be effective in cooperation with patterns (Gamma et all, 1995) or product line approaches (Clements, 2002) but these trends are out of scope of this paper.

Thus, to the present moment a wide experience of using the visual modeling in software engineering is accumulated and really, in many respects it has not justified the hopes. In this paper it is offered to revise the place of visual modeling in software process. For this purpose a new process model is introduced. It is based on an idea of conceptual field and division of the project activities into basic, conceptual, supplementary and correction.

The conceptual field is pointed out as the area of using the visual modeling in software project (independently of the diagrams type). In the paper visual modeling is considered as a way of thinking about software project problems (Quatrani, 1998). The principle difference of the approach presented here from others is a snapshot strategy that means we don't recommend creating information pieces that need to be supported during project life-cycle. We suggest some principles of operating diagrams and diagram-based techniques to dynamically manage the software project. Similar ideas are discussed in (Cockburn, 1999) but apart of the visual modeling and in the context of human aspect of software development process. One page method is similar to IDEF0 (IDEF0, 1993) (particular to author/commenter cycle), but concentrates on resolving some concrete problems instead of analysis the complicated area.

This approach accumulates author's experience of using visual modeling in different commercial and research projects in Saint-Petersburg State University and LANIT-TERCOM Corporation (Russia) for the last ten years. Author actively participated in one of them and investigated others. A lot of interesting information about visual modeling usage in software and business reengineering industry author obtained from feasibility studies performed in the IT-departments of some Russian banks.

2. Definitions

2.1 Visual modeling

Visual modeling is a usage of images in various business-fields (in the industry, science, management etc.). There are additional limitations on these images distinguishing them from arbitrary pictures - they are created from the standard "patterns" having defined semantics and way of usage. Such kind of limitations lead us to the semiotic approach and further to the visual modeling languages.

In a basis of visual modeling usage in software industry lies a simple observation that many people frequently draw pictures, when explaining a complex subject. Moreover the writing has appeared from the regulations of arbitrary images for the reproduction of exact sense. For example it was noticed that later ancient Egyptian hieroglyphs looked more like letters when the previous ones were more similar to the real objects described. So we see a fundamental role of the visual images and the visual channel of the person's perception for the creation of human mental world that is required to allow for people to do something together. This is due to the communicative effect of the visual modeling, permitting by means of images to transfer the information through the boundaries of the professional specialization, through the class and time barriers. So it is possible to make accumulation of the information and use it, for example, to control complex activities. The last makes attractive usage of visual modeling in management.

2.2 Conceptual Field of Project

In terms of conceptual field we shall understand some information assets of a project – its mission, main ideas and working schemes. The last means the empirically discovered rules assisting to control the project, and refinements of the main ideas arising during the project life-cycle. Speaking about operation with the conceptual field, we shall understand first of all operation with the working schemes, because the relations between the ideas and mission are hard to formalize.

The mission of the project is its place and role in a surrounding world. For example, the mission of the system of electronic document circulation is handle of a clerical work on a company, creation of transparent structure, increase of efficiency of business of the company.

The main ideas of the project are creative discoveries, due to which the project originates and is fulfilled successfully. They can be introduced from outside, can be created specially for the given project and/or during its life-cycle. For example, project business-idea is the answer to a question, why this software enterprise decides to develop system of an electronic document circulation: there are many kinds of such systems in the market. The reason is there was found a company ready to pay for it because of the uniqueness of its administration activities, where it is impossible to use existing systems, available in the market.

The working schemes are the detailed ideas of the project life-support mechanisms (for software projects it means process and product architecture). Process and product architecture may be strictly documented or not: it depends on the formalization style of the project. Finally the working schemes have to be presented in the minds of developers and should be clearly realized by managers and leading project technical staff. They have to be “alive” i.e. to be followed by the project. Working schemes can be reused from one project to another, but it always requires the customization because of the individuality of each software process.

2.3 Operating the conceptual field: long time and snapshot strategies

We will distinguish two different strategies of operating the conceptual field of the project: long time and snapshot.

The long time strategy means that the working schemes of the project have to be thoroughly elaborated and strictly followed by. But there are a lot of different work products in the software project – documents, specifications, etc., and a lot of efforts should be taken to support all of them in actual state. In this case, for example, design specification has to reflect all software architecture changes. And for each process policy change, the corresponding corrections of documentation have to be performed.

The snapshot strategy means that the only thing that needs to be clarified at the moment are formalized during project performing. Afterward these formalisms should be instantly used and after completion they have to be put into archive. In future no special efforts to keep them in the actual state are undertaken.

The main difference between long time and snapshot strategies is that in latter case formalization resolves a particular project's problems instead of achieving the goal to correspond some methodology. From this point of view the idea of separating two strategies of operating conceptual field is similar to "pull-push" theory (Zmud, 1984) widely used in the management.

The long time strategy takes a lot of resources applied for the whole conceptual field. Disadvantages of this approach are discussed in (Cockburn, 1999). But this approach is considered to be quite useful for the special parts of the conceptual field, because there are some situations where the supporting integrity is a way to manage the complex activities.

3. Conceptual field of software project

3.1 Place of conceptual field in software project: ideal model

Ideally the project activities should be divided into: conceptual field (adjustment of the basic management system of the project, elaboration of the software architecture principles, etc.), basis (coding, testing, etc.), supplementary and correction (to correlate the results of the conceptual field activities and the basic activities).

The supplementary and correction activities can be fulfilled permanently for the maintenance of corresponding the conceptual field and basic activities i.e. they support a sufficient level of the project integrity (gathering the project's information, performing some special integration activities – configuration management, testing, reporting, etc.).

Many participants of the project (architects, testers, etc.) can work with the separate parts of conceptual field. But it is the manager who oversees the conceptual field of the project – product architecture, the strategic goals of the project, political aspects (top management, customers), etc. The manager has the necessary overview, as he is not absorbed by some separate part of process and sees it as a whole.

3.2 Degree of a project unconceptuality

Unfortunately the software industry doesn't have any well-defined standard process. Moreover inside of every particular project there is a great variability of requirements, ways of implementations, etc. In this situation it's impossible to produce the general rules for achieving and supporting the project conceptual integrity. But for every project there is an empirical balance between: formal process assets (plans, policies, reviews, enterprise standards etc.) and success of the project; product architecture plan on one side and possibility and needs to change it on another, conceptual and "ad hoc" ways of carrying out the project.

The main question of the paper is how to direct the conceptual work from mind abstractions to an immediate practice. This is the problem of correct relations between project conceptual field and project activities. It is addressed to the project management. It is manager who decides whether it is possible to produce the next product version with patches instead of a correct solution that however takes time. But he can also find it better to miss this deadline and to implement the conceptual decision because the next deadlines can demand so many efforts that the project would be closed.

Degree of a project unconceptuality depends on specific project's parameters, such as its scope, technical and administration heterogeneity, kind of software (interesting software classification one can be found in (Jones, 1996), process model and technologies, financial state and perspectives of the project, etc.

3.3 Comparison with software methodologies

The suggested process model (basic project activities, conceptual field activities, supplementary and correct activities) is a special view on a particular software process and can be also applied to a software methodology. Below some of the well known methodologies are considered from the point of view of this model.

Almost the whole CMM[5] (Paulk et al, 1993) is in the conceptual field activities, supplementary and correction activities because its subject is a software process. Only software product engineering key process area may be considered as a set of the basic project activities. So establishing and supporting CMM in a project requires a lot of supplementary work. It could be effective in case of possible reuse of such kind of work.

Pairs phase/workflow of RUP (Kruchten, 1998) may be compared with suggested process model: for example, design on elaboration phase belongs to the conceptual field activities, but in a construction phase – to supplementary and correction activities. One can see a lot of conceptual field activities (mainly based on UML) but lack the real mechanisms for supplementary and correction activities. There is round trip problem for diagrams (and other software assets) and the code.

[5] Capability Maturity Model.

XP[6] (Kent., 1999) recommends not to perform the unnecessary conceptual work and is oriented to the balance of the project results (nearest) and project activities. Actually, the conceptual field exists implicitly in the minds of developers, when they apply this methodology. So this approach does not suit the big teams. The disadvantage of this approach is the absence of a border between conceptual and basic project activities. So this methodology can be applied only by experienced manager who should also be a technical leader of the project.

4. Visual modeling and operating the project conceptual field

We suggest usage of visual modeling in conceptual field activities. There were attempts to exploit it in the basic project activities, for example, as an alternative to common programming (Raeder, 1985), but they were not successful. One of the possible reasons of the failure is that the specifications on programming languages are intended not only for human, but also for computers and cannot be expressed in a compact graph view (which would let operating the text on programming language by means of diagrams)[7]. The experience of software development has shown that it is convenient to visualize the essence of the complex subject, omitting numerous details.

4.1 The long time strategy

One particular case of a long time strategy should be noticed. It is very useful, when the main project's diagram (one or the set of such diagrams) exists. It may be used for the presentation purposes and for introducing new people into the running project. Such diagrams have to be good. Once created, they exist for a long time. The author observed architecture diagram of a telecommunication system (28 units and a lot of links) being used in demonstrational purpose for many years. This diagram even outlasted the project itself.

The initiative in creating such diagrams usually comes from managers, because developers don't need them – everything is clear to them, and if something is not clear, they would prefer the traditional communication.

These diagrams depict main ideas and working schemes of the project with large granulation scale and it is not hard to correct them. One more reason why it is easy is that there are a few of such diagrams.

4.2 The snapshot strategy

Here are the principles of operating diagrams by snapshot strategy:

- **Diagrams should have a context.** Author experienced many situations (during business-processes specifications, usage of reverse engineering tools, etc.) where creation UML diagrams was an end in itself. The creators were not aware of the aim of

[6] Extreme Programming.

[7] There are some exceptions from this rule (database scheme and reactive systems (Koznov, 2000) modeling), but in these cases there is a problem of integration of the software presented by visual modeling and by general way.

the work but through they had to do as many diagrams as they could. But diagram presence does not automatically guaranties the clarifying of the complex domain area. It needs some context. For example diagrams look naturally inserted into some text (report, description etc.). But this is not enough, the diagrams should be "good".

- **What does a "good" diagram mean?** Such kind of a diagram has to produce integral impression. In particular it means that there should not be a lot them. One diagram has to contain multiple information levels to have possibility to return to the diagram form a different discussion points, to refer to it from the various parts of a report, etc. Numerous ways ought to lead to the same diagram. Each new view of a complicated subject should not be depicted in the separate diagram but to be demonstrated on the same one. Otherwise there will be so many diagrams, they will change so fast each other that it will become impossible to understand anything. For example this is a typical situation for Power-Point presentations. But "good" diagrams lay at the discussion table, hang on the wall in the manager's room, etc. The idea of multiple views (UML, 2001) is more applicable for the same subject and different information consumers.
- **Diagrams should be intuitionaly obvious.** From one side such kind of diagrams should be created on a well-defined and wide-known diagram language. But the standard language doesn't permit some familiarities which are able to make the diagram much more clear and expressive. And UML extension mechanism (UML, 2001) and graphic editors customization facilities do not help because of the unpredictability of such kinds of particularities. That is why graphic editors like Visio are often used instead of CASE-tools[8]. CASE-tools are more preferred when using visual modeling in basic activities of project (in case of detailed design of database applications, modeling reactive systems) because in this situation we have to create a lot of diagrams and would like to have some special facilities to operate them.

4.3 One page method

This method was used in feasibility studies of complicated software projects, for research projects, when supervising students thesises and preparing seminars, lectures, training, etc.

One or more project's participants work with an expert to formalize and integrate project's assets – actually to make a snapshot of project conceptual field. There are some points when the project looses orientation and has a chance to fail without such revision.

A set of meetings are conducted. At first the project problem is identified and after that is elaborated more and more and is transformed to decision. This jump happens suddenly and is stimulated by the expert. The main idea of the method is frequent meetings (from several times per one day to the one per 2-3 days) and concentrated work around the one specification. Every time the same page with text and/or diagram is discussed and after that corrected and discussed again. This page contains the scheme, the essence of understanding

[8] Actually Microsoft stopped integrating Developer Studio with Rational Rose based CASE-tool. At the moment it develops its own graphic editor on the base of purchased Visio.

the problem (with blank spots) that turn into decision scheme, working plan, etc. The iteration is completed when the conceptual problem is resolved and can be implemented (without expert). So all work is carried out in the conceptual field of the project. This iteration can take from one week to several months.

It is desirable to use diagrams in such kinds of specifications. They increase its expressive power and contribute to the appearance of the fruitful associations of the clients. As a rule one small diagram is enough. However it is possible for the specification to contain more diagrams in case if its size is more than one page because of some peculiarities of a problem domain, or clients, or project problems. All specification also may consist of the only diagram with pieces of text inserted in graphic elements.The use of the following diagrams is productive in visual modeling in this case: SA[9] (Ross, 1977), entity-relationship diagrams (Chen, 1976), data-flow diagrams (Iordan, 1989), functional diagrams (Z.100, 1993).

5. Conclusions

The approach offered in this paper revises the visual modeling position in software process from the perspectives of management. The usage of the visual modeling in management is not new idea: for example, SADT[10] suggested in 70th.was widely used in the complex industrial projects (not only software ones!). Unfortunately at the last time the visual modeling is developed in software industry mainly from the perspective of analysis and design usage.

In order to clearly determine the place of visual modeling in software project, the notion of the conceptual field is introduced. After that the usage of visual modeling in management is separated from its' usage in software analysis and design. The snapshot strategy is suggested as a main way of operating visual modeling in management. One page method as a one possible implementation of the snapshot strategy is considered.

6. References

W. Royce (1998). Software Project Management. A Unified Framework. Addison-Wesley.

F.P.Jr.Brooks (1975) The Mythical Man-Month. - S.L.: Addison-Wesley.

M.C.Paulk, B.Curtis, M.B.Chrissis, C.V.Weber (1993). Capability Maturity Model for Software, Version 1.1. Technical Report CMU/SEI-93-TR-024 ESC-TR-93-177.

W.Humphrey (1990) Managing the Software Process. Addison-Wesley.

G.Raeder (1985). A Survey of Current Graphical Programming Techniques. IEEE Computer. Pp. 11-25.

D. Koznov (2000), Visual modeling of component software. (in Russian) Ph.D. Thesis, Saint-Petersburg State University.

[9] Structured Analysis
[10] Structured Analysis Design Technique

P.P.Chen (1976) The entity-relational model. Toward a unified view of data //ACM TODS, no. 1, pp. 9-36.

ITU Recommendation Z.100 – Appendices I and II: SDL Methodology Guidelines, SDL Bibliography. (1993) 129 p.

I.Jacobson (1992). Object-Oriented Software Engineering. ASM press. 1992, 528 p.

J.Cappers (1996). Patterns of Software System Failure and Success. International Thomson Computer Press, Boston, Massachusetts.

OMG Unified Modeling Language Specification. Version 1.4 (2001). http://www.omg.org

M.Fauler (2001). The New Methodology. Available online at http://martinfowler.com/articles/newMethodology.html

K.Beck(1999).Extreme Programming Explained: Embrace Change Addison-Wesley. 224 p.

M.Fowler, K.Scott (1999). UML Distilled: Applying the Standard Object Modeling Language. Addison-Wesley, 185 p.

P.Clements, L.M.A.Northrop (2002). Framework for Software Product Line Practice. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

E.Gamma, R.Johnson, R.Helm, J.Vissides (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995, 395 p.

P. Kruchten (1998). The Rational Unified Process: An Introduction.Addison-Wesley. 255 p.

T.Quatrani. (1998). Visual Modeling with Rational Rose and UML. Addison-Wesley.

A.Cockburn (1999). Characterizing People as Non-Linear, First-Order Components in Software Development. Humans and Technology Technical Report, TR 99.05.

Integration Definition For Information Modeling (IDEF1X) (1993). Draft Federal Information Processing Standards Publication 184, 1993, 87 p.

R. W.Zmud (1984). An Examination of 'Push-Pull' Theory Applied to Process Innovation in Knowledge Work, *Management Science*, Vol. 30(6), 1984, pp. 727-738.

E.Yourdan (1989). Modern Structured Analysis. Yordan Press, 672 p.

D.T.Ross (1977). Structured Analysis: A language for Communication Ideas. IEEE Transaction on Software Engineering. Vol SE-3, N 1, pp. 16-34.