Software Engineering and Development

SOFTWARE DEVELOPMENT PROCESS DYNAMICS MODELING AS STATE MACHINE

Leonid Lyubchyk, Vasyl Soloshchuk

***Abstract**: Software development process modeling is gaining increasing interest among academic researchers and IT-industry practitioners alike as an approach for analyzing complex business questions of software production from the strategic management of software development, to supporting process improvements. A number of simulation models of software dynamics have been developed based on system dynamic approach with the use of special simulation programming tools. At the same time analytical software development modeling are of great interest as a tool of process optimization and control. In this paper the software development processes model based on the state machine and queuing network approach is proposed. The goal is the software development process dynamics modeling t hat is the evaluation of software product readiness progress in the course of time. Software product is considered as a set of system requirements, and software development process is represented as consecutive transition of each system requirement through the states of the defined state machine. The state machine model is a queuing network with service nodes which correspond to a number of implementers of system requirements on each stage of the development. Implementers are business analytics, architectures, developers, testers, etc. In general, the state machine is defined according to the used software development methodology. The markovian processes theory is further used for the queuing network modeling and quality indexes estimation. The proposed model can be used as a basis for modeling the dynamics of a software project development to get estimations of the following project parameters: effort, duration, cost, resources usage, development schedule, etc.*

***Keywords**: software development process dynamics, software engineering, production modeling, state machine, project management*

***ACM Classification Keywords**: D. Software – D.2 SOFTWARE ENGINEERING – D.2.8 Metrics (Process metrics, Product metrics), D.2.9 Management (Cost estimation, Life cycle, Productivity, Software process models, Time estimation); K. Computing Milieux – K.6 MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS – K.6.1 Project and People Management (Life cycle, Staffing), K.6.3 Software Management (Software process), K.6.4 System Management (Management audit)*

Introduction

Currently, the problem of estimating the parameters of software development projects using different mathematical and statistical techniques is relevant, because it allows minimizing uncertainty in the planning, which significantly reduces the risk of project failure [McConnell, 2006]. However, most models estimating parameters of software development projects are static and do not allow to estimate progress dynamics of project. To analyze the dynamics of the project progress such methods as system dynamics and simulation modeling are applied [Abdel-Hamid, Madnick, 1991], [Lakey, 2004], [Madachy, 1996]. For modeling the dynamics of software development the development models of complex engineering product by the engineering team are also used [Soo-Haeng Cho, Eppinger, 2005]. The product itself, team parameters and the process of development hold many uncertainties. To cover all the processes of the project both the effort that brings to the increase of the developed product (programming, documentation) and work that do not directly contribute to increasing the size of the product (testing, control) should be taken into account [Han, Lee, Fard, Pena-Mora, 2007]. Mathematical model for analyzing the dynamics of the project progress must be practically applicable, i.e. it must take into account real-world objects and processes that can be clearly identified and measured in a real project. Classical methods of identification of operations models can be used for these purposes [Pervozvansky, 1975]

Structure of software development project

There are many methodologies for software development, but there are a number of basic objects, which are present in the project structure for any methodology [Sommerville, 2001].

Software product requirements. There are different levels of detail and description of the requirements for a software product, namely the business requirements, user requirements, functional and nonfunctional requirements. Functional and nonfunctional requirements are essential for further usage in the project. One of the most convenient requirements classifications for a software product is a FURPS model, which means:

- Functionality – feature set, capabilities, generality, security;
- Usability – human factors, aesthetics, consistency, documentation;
- Reliability – frequency/severity of failure, recoverability, predictability, accuracy, mean time to failure;
- Performance – speed, efficiency, resource consumption, throughput, response time;
- Supportability – testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, localizability, portability;

Project and process requirements. Project requirements are development process requirements, i.e. what methodology will be used, what is the structure of development stages, the length of iterations, the structure of roles in the project, the distribution of effort between the various processes and stages of the project, etc. This also includes the definition of project requirements, i.e. how the developer will perform the target system creation tasks.

Work Breakdown Structure (WBS). WBS – hierarchical work decomposition conformed to distribution results. This work decomposition should be performed to achieve the objectives of the project and agreed distribution results delivery [PMBOK, 2004]. With the help of hierarchical work structure the entire contents of the project is structured and defined. WBS divides the project to smaller and more manageable parts, where each lower level of WBS provides a more detailed definition of project works. For planned operations conformable to the lower-level WBS elements (they are called work packages), time schedule and cost estimates can be determined as well as operations monitor and control can be performed.

Team structure and resources calendar. The structure of the team specifies what resources, i.e. what specialists are available for project performing. Each task needs its own kind of resources. Resources classification may vary for different software development methodologies. It should be noted that each single resource can combine multiple roles and, accordingly, can be used to perform various types of operations. Resources calendar is the calendar of working days and days off that determines dates in which each resource may or may not be used. Usually it includes working and nonworking periods of each resource [PMBOK, 2004].

Project time schedule includes a calendar schedule of project stages performance in accordance with WBS and availability of resources from the resource calendar. In this case the following parameters for the work should be stated: sequence number of work according to WBS, name of work in accordance to WBS, resource assigned for the job performance, the duration of performance, date and time of starting the work and its completion, previous and subsequent operations. Also, the time schedule states the milestone of the project. Project time schedule can be presented as a table or using Gantt charts (sometimes a term horizontal bar chart is used) [PMBOK, 2004].

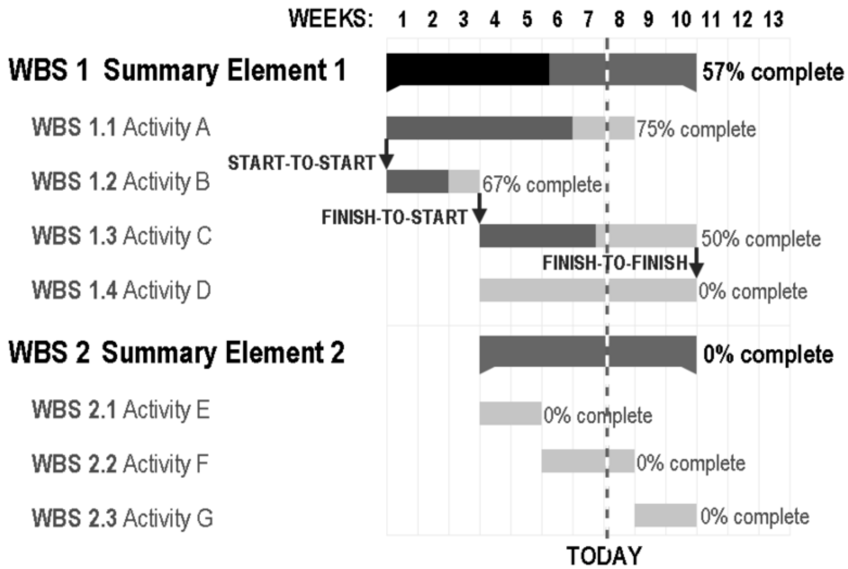

Figure 1. A Gantt chart showing three kinds of schedule dependencies (in red) and percent complete indications

Problem definition

To simulate the dynamics of the project progress the structure of the project should be formally described on the basis of mentioned above elements, which are included in the structure of the software development project. The project structure should also include a model of product requirements transition from one state of product readiness to another. Transitions from one state of product requirements readiness to another depend on the methodology of project development.

Parameters and elements of mathematical model of software development project

In accordance with the described project structure following formal objects and project parameters description can be introduced.

Objects to describe the structure of process requirements. These objects should take into account specificity of development methodology, which is used in a simulated project:

- L – set of operations (types of work), which obtain in the selected development methodology;
- S – set of states of product requirements readiness;
- $W = \{w_{ij}\}$ – matrix with the types of work for the transition of requirements between the states;
- w_{ij} – type of work to transit requirements from i to j state, where $i \in S$, $j \in S$, $w_{ij} \in L$;
- $P = \{p_{ij}\}$ – probability of transition between states matrix;
- p_{ij} – probability of transition from i to j state, where $i \in S$, $j \in S$;
- $\sum_{i \in S} p_{ij} = 1$ – condition for the completeness exit from any state;

- *A* – set of artifacts that can be developed during the project.

Example of set of works L = {1, 2, 3, 4, 5, 6, 7, 8}, where:

- 1: specified requirements;
- 2: software architecture design (business logic, user interface, data and database structure);
- 3: programming;
- 4: quality assurance / testing;
- 5: sending requirements for revision, defect description
- 6: approval of programmed requirement;
- 7: correction of detected defects;
- 8: version release, which will include the requirement;

Example of set of readiness states S = {1, 2, 3, 4, 5, 6, 7, 8}, where:

- 1: identified requirement (found but not yet described in detail);
- 2: specified requirement (analyzed and included into requirements specification of developing software product);
- 3: designed requirement (included into developing software product architecture description);
- 4: programmed requirement (programmed and included into the current working software product version);
- 5: tested requirement (checked for defects and compliance with requirements included into developed software product);
- 6: defective requirement (defects identified in functioning or incorrect requirements implementation);
- 7: approved requirement (has been tested and approved for inclusion into the current product version);
- 8: released requirement (included into formal software product package).

Transitions between states, which are set by matrix *W*, are defined by using development methodology and may be represented as a network. An example of such a network is shown in Fig. 2.

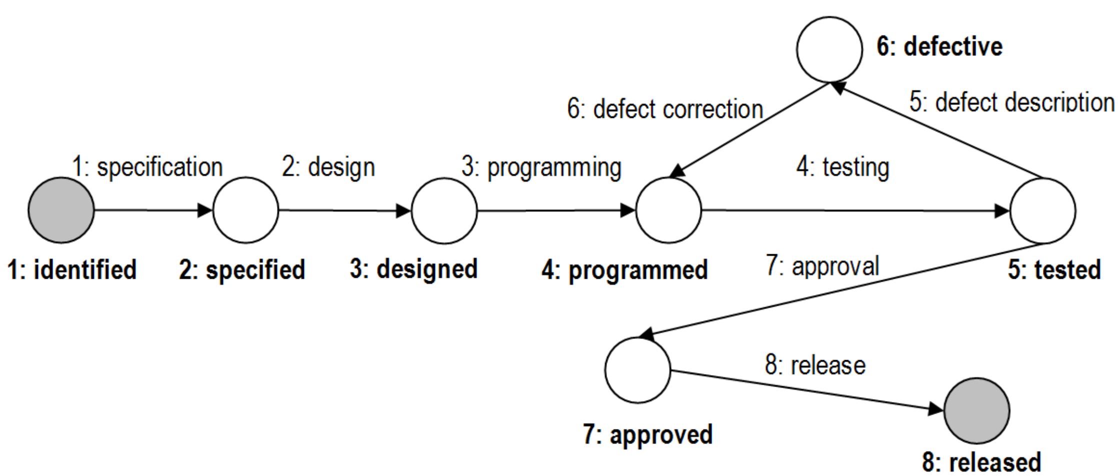

Figure 2. Network of transitions between software product requirements states

Objects to describe structure of product requirements. Let us introduce the notation to describe the parameters of the software product requirements:

- *Q* – set of requirement types;
- *R* – set of software product requirements;
- M_r – set of requirements that must be implemented before the start of the *r*-th requirement, $r \in R$;
- $A = \{b_r\}$ – set the types for each requirement, $r \in R$;
- a_r – type of *r*-th requirement, $r \in R$, $a_r \in Q$;
- $B = \{b_r\}$ – set with the current state of readiness for each requirement, $r \in R$;
- b_r – state of readiness for *r*-th requirement, $r \in R$, $b_r \in S$;
- z_r – size of *r*- th requirement, $r \in R$;
- c_r – priority of implementing of the *r*-th requirement, $r \in R$.

Objects to describe resources and resources calendar. Let us introduce the notation for describing the resources parameters:

- *K* – set of available resources;
- $V_k = \{v_{kql}\}$ – matrix of productivity for *k*-th resource;
- v_{kql} – productivity performance when *l*-th work for *q*-th requirement type is done using *k*-th resource, where $q \in Q$, $k \in K$, $l \in L$.

If the *k*-th resource is not able to perform *l*-th type of work for the *q*-th requirement type, then: $v_{kql} = 0$. If the *k*-th resource is able to perform *l*-th type of work for the *q*-th requirement type, then the function of productivity performance is a complex function of many variables:

$$v_{kql} = v(k, q, l, z, \ldots) \tag{1}$$

where *z* – size of requirement.

Available time intervals for performing the work of *k*-th resource:

- T_k – set of initial moments of the working periods of time;
- D_k – set working time intervals duration.

Project schedule. Project schedule is a schedule of requirements transfer from initial states to final states with fixation of specific values for all the probability variables and uncertainties.

The basis for project schedule is time required for each work to perform requirements transition from one state to another. The duration of *r*-th requirement transition from *i*-th state to *j*-th state can be set with following functions:

$$d_{rij} = \frac{z_r}{v(k, q, w_{ij}, z_r, \ldots)} \tag{2}$$

where $i \in S$, $j \in S$, $q \in Q$, $k \in K$, $w_{ij} \in L$,

z_r – size of requirement for r-th requirement, $r \in R$,

k – number of the resource, which performs this work;

q – requirement type;

w_{ij} – type of work to transit i-th state to j-th state.

Project schedule should also include estimation of the sequence of requirements state changes, taking into account the transition probability matrix *P*.

Constructing technique for model of project progress dynamics

Solution of this problem should include:

- estimation of project parameters: duration, effort costs, resource costs, quality level;
- estimation of resource needs and their time loading;
- estimation of project schedule;
- estimation of dynamics in the state of requirements readiness transition and estimation of requirements size in each state.

Formal project structure model described above can be used to solve these problems. However, this requires completion of the model with the following features:

- select methods and algorithms for modeling;
- develop method to estimate resources productivity, i.e. method of $v_{kql} = v(k, q, l, z, \ldots)$ function constructing;
- develop function of dynamic changes in the set of requirements R with time R = R (t).

The estimation of each option is best to give as a series of values indicating the confidence level of each value, i.e. probability of achieving such values. Fig. 3 shows the chart estimating date of project completion, which depends on the possibilistic project duration. Similar valuations should be issued for effort, cost, quality and other parameters.

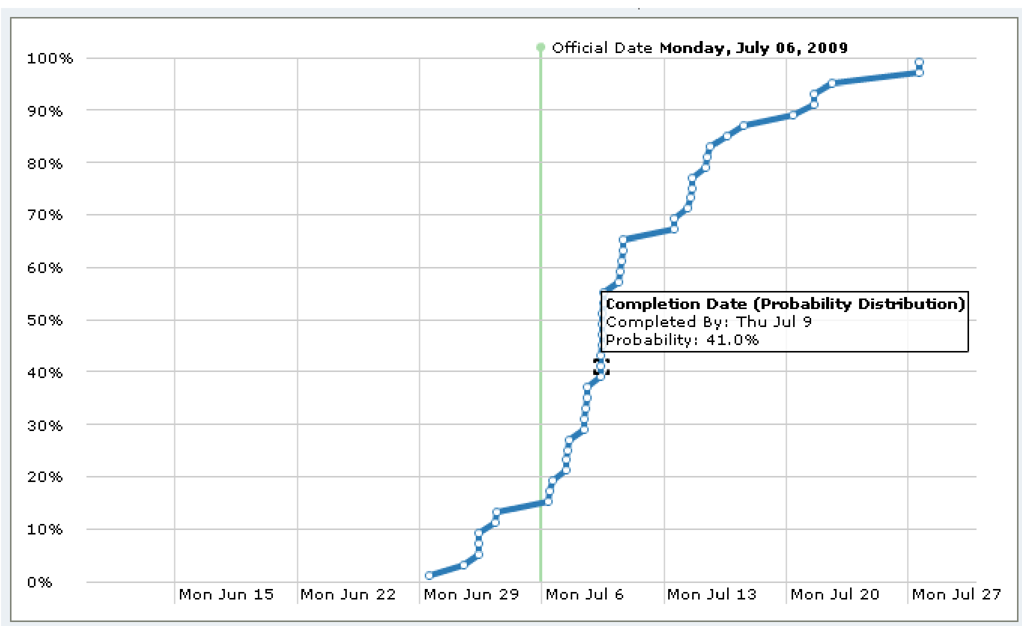

Figure 3. Example of estimating completion date (Probability Distribution)

Solution of project progress dynamics modeling problem will also give an opportunity to estimate requirements states transition dynamics and estimate requirements size in each of the states in the form presented in Fig. 4.

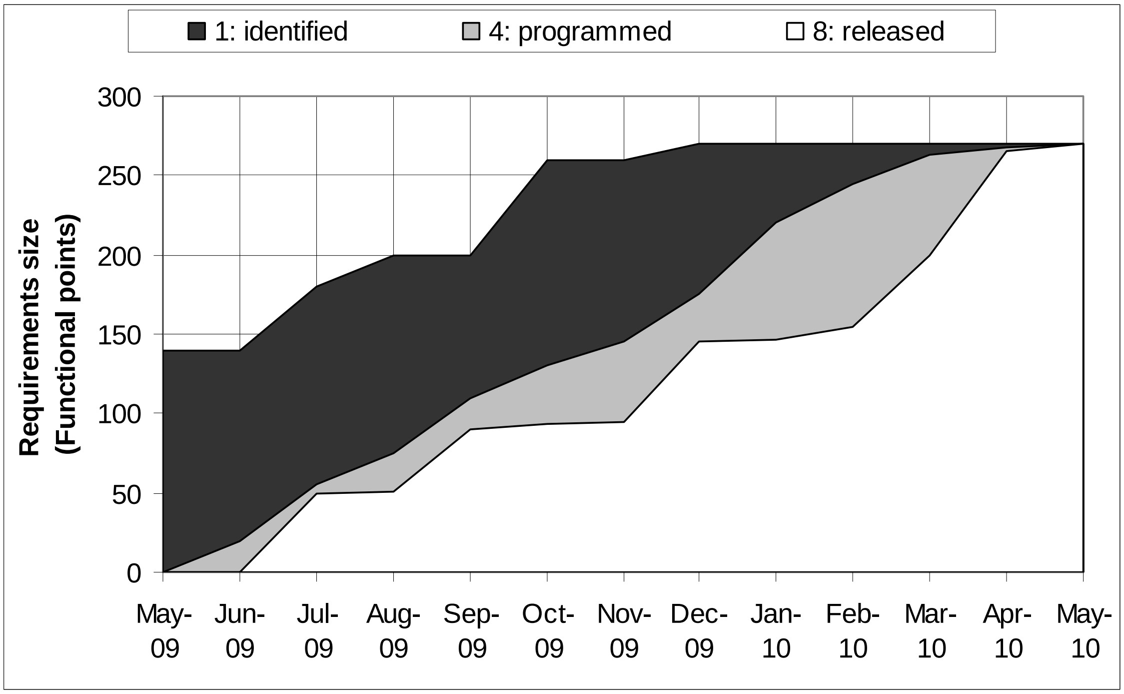

Figure 4. Dynamics of requirements size changing in different readiness states

Conclusion

This article reflects a method of formal description of the model of the software development project structure. In addition, problem of forecasting the dynamics of the project progress was formulated. The problem can be solved using developed project model. Developed model includes software product requirements state transition model. From mathematical point of view, we have got a model that has several important properties:

- software product requirements are objects that may have several states;
- objects state varies according to stated algorithm from generating state to the absorbing state;
- transition from one state to another is stated by certain transition probability;
- object always possesses some state for a period of time;
- new requirements can occur with the lapse of time, i.e. new objects can be generated;
- process is finite.

Mentioned properties allow to state that this process can be described by Markovian process model. In future we plan to use simulation techniques of Markovian process to solve stated problems of modeling project progress dynamics.

Bibliography

[McConnell, 2006] McConnell. S. Software Estimation: Demystifying the Black Art. – Redmond, Wa.: Microsoft Press, 2006. – 352 p.

[Abdel-Hamid, Madnick, 1991] Abdel-Hamid T., Madnick S.E. Software Project Dynamics: an Integrated Approach. – Prentice-Hall Software Series, Prentice-Hall Publishing Co., Englewood Cliffs, NJ, 1991 – 264 p.

[Lakey, 2004] Lakey P. A Hybrid Software Process Simulation Model for Project Management // Proceedings of ProSIM 2004 Conference, prosim.pdx.edu – 2004.

[Madachy, 1996] Madachy R.J. System dynamics modeling of an inspection-based process // Software Engineering, 1996., Proceedings of the 18th International Conference on 25-29 Mar 1996. P. 376-386.

[Soo-Haeng Cho, Eppinger, 2005] Soo-Haeng Cho, Eppinger S.D. A simulation-based process model for managing complex design projects // Engineering Management, IEEE Transactions. Volume: 52. Issue: 3, Aug. 2005. P.316- 328.

[Han, Lee, Fard, Pena-Mora, 2007] Sangwon Han, Sang Hyun Lee, Fard M.G., Pena-Mora F. Modeling and representation of non-value adding activities due to errors and changes in design and construction projects // Simulation Conference, 2007 Winter. 9-12 Dec. 2007. P. 2082-2089.

[Pervozvansky, 1975] Pervozvansky A.A. Mathematical models in production management. – "Nauka" publishing house, Moscow, 1975. – 616 p.

[Sommerville, 2001] Sommerville I. Software Engineering, 6-th edition. – Harlow: Addison Wesley Longman, 2001. – 720 p.

[PMBOK, 2004] A Guide to the Project Management Body of Knowledge (PMBOK® Guide), 3rd edition. – Project Management Institute, Pennsylvania, USA, 2004. – 401 p.

Authors' Information

Leonid M. Lyubchyk *– Professor, Head of Computer Mathematics and Mathematical Modeling Department, National Technical University "Kharkiv Polytechnic Institute"; Frunze Str. 21, 61002 Kharkiv, Ukraine; e-mail: lml51@mail.ru*

Major Fields of Scientific Research: Control theory (robust and model-based predictive control, adaptive and learning systems), System modeling and simulation, Financial time-series modeling and forecasting, Computational intelligence for data analysis

Vasiliy M. Soloshchuk *– PhD Student, Computer Mathematics and Mathematical Modeling Department, National Technical University "Kharkiv Polytechnic Institute"; Frunze Str. 21, 61002 Kharkiv, Ukraine; e-mail: vsolo@insart.com*

Major Fields of Scientific Research: Software engineering, Production modeling, Project management, System modeling and simulation