

Л. М. ЛЮБЧИК, д-р техн. наук, профессор НТУ «ХПИ»,
В. М. СОЛОЩУК, аспирант НТУ «ХПИ»

МОДЕЛИРОВАНИЕ ДИНАМИКИ РАЗВИТИЯ ПРОЕКТА ПО РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ОСНОВЕ ОЦЕНКИ СОСТОЯНИЙ ГОТОВНОСТИ ПРОДУКТА

В статті розглянуто модель станів готовності вимог до програмного продукту, яка може бути використана як основа для моделювання динаміки розвитку проекту. Описано формальну модель структури проекту по розробці програмного забезпечення. Зроблено допущення про можливість використання теорії марківських процесів для моделювання через схожість процесів.

В статье рассмотрена модель состояний готовности требований к программному продукту, которая может быть использована для моделирования динамики развития проекта. Описана формальная модель структуры проекта по разработке программного обеспечения. Сделано предположение про возможность использования теории марковских процессов для моделирования из-за схожести процессов.

The states model of software product requirements is described in the paper. This model can be used as a basis for modeling the dynamics of a project development. The formal model of software project structure is described. The assumption is made that the theory of markovian processes can be used for the further modeling because of the similarity of the processes.

Введение. В настоящее время задача оценки параметров проектов по разработке программного обеспечения (ПО) с применением различных математических и статистических методов является актуальной, т.к. позволяет минимизировать неопределенность в планировании, что существенно уменьшает риск провала проекта [1]. Однако большинство моделей оценки параметров проекта по разработке ПО являются статичными и не позволяют оценить динамику развития проекта. Для анализа динамики развития проекта применяются методы системной динамики и имитационного моделирования [2-4]. При моделировании динамики проекта по разработке ПО также используют модели, которые существуют для опытно-конструкторских проектов, которые заключаются в разработке сложного инженерного продукта командой инженеров [5]. При этом имеет место множеством факторов неопределенности в самом продукте, параметрах команды и самом процессе разработки. Для охвата всех процессов проекта необходимо учитывать работы, которые приносят вклад в увеличение разрабатываемого продукта (программирование, разработка документации), и работы, которые не вносят непосредственного вклада в увеличение размера продукта (тестирование, управление) [6]. Математическая модель для анализа динамики развития проекта должна быть практически применима, т.е. она должна учитывать реальные объекты и процессы, которые можно четко идентифицировать и измерить в реальном проекте. Для этого можно использовать классические методы идентификации моделей операций [7].

Структура проекта по разработке ПО. Существует множество методологий разработки ПО, однако существует ряд основных объектов, который присутствует в структуре проекта для любой методологии [8].

Требования к программному продукту. Различают разные уровни детализации и описания требований к программному продукту, а именно бизнес-требования, пользовательские требования, функциональные и нефункциональные требования. Функциональные и нефункциональные требования являются основными для дальнейшего использования в проекте. Одной из наиболее удобных классификации требований к программному продукту является модель FURPS, что означает:

- Functionality (Функциональность);
- Usability (Применимость);
- Reliability (Надежность);
- Performance (Производительность);
- Supportability (эксплуатационная пригодность);

Требования к проекту и процессу. Под требованиями к проекту понимаются требования к процессу разработки, т.е. к тому, какая методология будет использована, какова структура этапов разработки, длина итераций, структура ролей в проекте, распределение трудозатрат между различными процессами и стадиями в проекте и т.д. Сюда же относятся вопросы формулирования требований к проекту, т.е. к тому, как разработчик будет выполнять работы по созданию целевой системы.

Иерархическая структура работ (ИСР, или Work Breakdown Structure, WBS). ИСР – это согласованная с результатами поставки иерархическая декомпозиция работ, которые команда проекта должна выполнить для достижения целей проекта и создания оговоренных результатов поставки [9]. С ее помощью структурируется и определяется все содержание проекта. ИСР подразделяет работы проекта на более мелкие и более управляемые части, где на каждом более низком уровне ИСР дается более детальное определение проектных работ. Для запланированных работ, соответствующих элементам низшего уровня ИСР (их еще называют пакетами работ), можно определять график выполнения, сметную стоимость, осуществлять наблюдение и контроль над ним.

Структура команды и календарь ресурсов. Структура команды задает, какие ресурсы, т.е. какие специалисты, имеются для выполнения проекта. Для выполнения каждой задачи необходим свой вид ресурсов. Классификация ресурсов может различаться для разных методологий разработки. Необходимо отметить, что один ресурс может совмещать в себе несколько ролей и, соответственно, может быть задействован для выполнения различных видов работ. Календарь ресурсов является календарем рабочих и нерабочих дней, определяющий, в какие даты каждый ресурс может или не

может быть использован. Обычно содержит определенные рабочие и нерабочие периоды каждого ресурса [9].

Календарный план проекта включает в себя календарное расписание выполнения работ по проекту в соответствии с ИСР и с учетом доступности ресурсов из календаря ресурсов. При этом указываются следующие параметры для работ: номер работы в соответствии ИСР, название работы в соответствии ИСР, выделенный ресурс для выполнения данной работы, длительность выполнения работы, дата и время начала работы, дата и время окончания работы, предшествующие работы, последующие работы. Также в плане указываются ключевые точки выполнения проекта. Календарный план проекта может быть представлен как в табличном виде, так и с использованием диаграммы Ганта (Gantt Chart, также иногда употребляют термин *столбиковая горизонтальная диаграмма*) [9].

№	Название задачи	Начало	Завершение	Продолжительность	2004				2005						
					01	02	03	04	01	02	03	04			
1	Фаза 1 - ПОДГОТОВКА	01.01.2004	24.03.2004	60d	█	█	█	█	█	█	█	█	█	█	█
2	Нанять консультант	01.01.2004	21.01.2004	15d	█	█	█	█	█	█	█	█	█	█	█
3	Разместить центр данных	22.01.2004	18.02.2004	20d	█	█	█	█	█	█	█	█	█	█	█
4	Разместить филиалы	19.02.2004	10.03.2004	15d	█	█	█	█	█	█	█	█	█	█	█
5	Активировать ISDN и службу Frame Relay	11.03.2004	24.03.2004	10d	█	█	█	█	█	█	█	█	█	█	█
6	ФАЗА 2 - РАЗВИТИЕ	25.03.2004	26.05.2004	45d	█	█	█	█	█	█	█	█	█	█	█
7	Установить службы VAN	25.03.2004	14.04.2004	15d	█	█	█	█	█	█	█	█	█	█	█
8	Увеличить загрузку EDI	15.04.2004	26.05.2004	30d	█	█	█	█	█	█	█	█	█	█	█
9	Перенести центры управления запасами	27.05.2004	18.08.2004	60d	█	█	█	█	█	█	█	█	█	█	█
10	ФАЗА 3 - Установка связи	19.08.2004	29.12.2004	95d	█	█	█	█	█	█	█	█	█	█	█
11	Сократить офисы региональных торговых представителей	19.08.2004	20.10.2004	45d	█	█	█	█	█	█	█	█	█	█	█
12	Открыть офисы для продавцов	21.10.2004	29.12.2004	50d	█	█	█	█	█	█	█	█	█	█	█
13	ФАЗА 4 - СЕЗОННЫЙ ПЕРСОНАЛ	30.12.2004	25.05.2005	105d	█	█	█	█	█	█	█	█	█	█	█
14	Организовать работу сезонных офисов	30.12.2004	19.01.2005	15d	█	█	█	█	█	█	█	█	█	█	█
15	Открыть дополнительные офисы для продавцов	20.01.2005	02.03.2005	30d	█	█	█	█	█	█	█	█	█	█	█
16	Завершить проект Service 21	03.03.2005	25.05.2005	60d	█	█	█	█	█	█	█	█	█	█	█

Рис. 1. Пример диаграммы Ганта

Постановка задачи. Для моделирования динамики развития проекта необходимо формально описать структуру проекта на основе перечисленных выше элементов, которые входят в структуру проекта по разработке ПО. Структура проекта также должна включать в себя модель перехода требований к продукту из одного состояния готовности продукта к другому. Переходы из одного состояния готовности требований к продукту к другому состоянию готовности обусловлены методологией разработки проекта.

Параметры и элементы математической модели проекта по разработке ПО. В соответствии с описанной структурой проекта введем следующее формальное описание объектов и параметров проекта.

Объекты для описания структуры требований к процессу. Данные объекты должны учитывать специфику методологии разработки, которая используется в моделируемом проекте:

- L – множество видов работ (операций), которые существуют в выбранной методологии разработки;
- S – множество состояний готовности требований к продукту;
- $W = \{w_{ij}\}$ – матрица с типами работ для перевода требований между состояниями;
- w_{ij} – тип работы для перевода требования из состояния i в состояние j , где $i \in S, j \in S, w_{ij} \in L$;
- $P = \{p_{ij}\}$ – матрица вероятности переходов между состояниями;
- p_{ij} – вероятность перехода из состояния i в состояние j , где $i \in S, j \in S$;
- $\sum_{i \in S} p_{ij} = 1$ – условие полноты выхода из любого состояния;
- A – множество артефактов, которые могут быть разработаны в ходе проекта.

Пример множества видов работ $L = \{1, 2, 3, 4, 5, 6, 7, 8\}$, где:

- 1: специфицирование требований;
- 2: проектирование архитектуры программного продукта (бизнес-логика, пользовательский интерфейс, структура данных и базы данных);
- 3: программирование;
- 4: проверка качества / тестирование;
- 5: отправка требования на доработку, описание дефекта
- 6: одобрение запрограммированного требования;
- 7: исправление обнаруженных дефектов;
- 8: выпуск версии, в которую будет включено требование;

Пример множества состояний готовности $S = \{1, 2, 3, 4, 5, 6, 7, 8\}$, где:

- 1: выявленное требование (выявлено, но еще не описано детально);
- 2: специфицированное требование (проанализировано и включено в спецификацию требований к разрабатываемому ПО);
- 3: спроектированное требование (включено в описание архитектуры к разрабатываемому ПО);
- 4: разработанное требование (запрограммировано и включено в текущую рабочую версию ПО);
- 5: протестированное требование (проверено на дефекты и соответствие требованиям в составе запрограммированного ПО);
- 6: дефектное требование (выявлены дефекты в функционировании или имеет место неправильная реализация требования);
- 7: одобренное требование (прошло тестирование и одобрено к включению в текущую версию продукта);

- 8: выпущенное требование (включено в официальную сборку ПО).
- Переходы между состояниями, которые задаются матрицей W , определяются используемой методологией разработки, и может быть представлено в виде сети. Пример такой сети представлен на рис. 1.

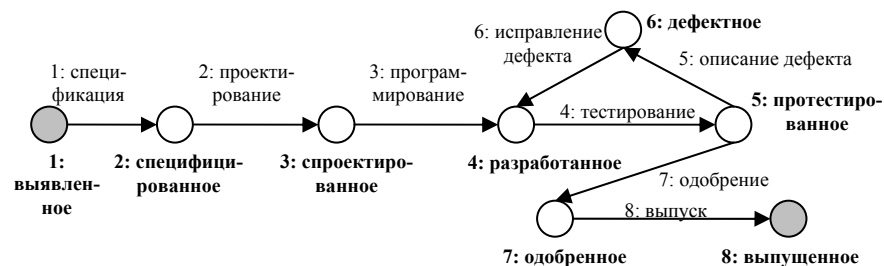


Рис. 2. Сеть переходов между состояниями требования к программному продукту

Объекты для описания структуры требований к продукту. Введем обозначения для описания параметров требований к программному продукту:

- Q – множество типов требований;
- R – множество требований к разрабатываемому программному продукту;
- M_r – множество требований, которые необходимо реализовать до начала реализации r -го требования, $r \in R$;
- $A = \{b_r\}$ – множество с типами по каждому требованию, $r \in R$;
- a_r – тип r -го требования, $r \in R, a_r \in Q$;
- $B = \{b_r\}$ – множество с текущим состоянием готовности по каждому требованию, $r \in R$;
- b_r – состояние готовности r -го требования, $r \in R, b_r \in S$;
- z_r – размер r -го требования, $r \in R$;
- c_r – приоритет реализации r -го требования, $r \in R$.

Объекты для описания ресурсов и календаря ресурсов. Введем обозначения для описания параметров ресурсов:

- K – множество доступных ресурсов;
- $V_k = \{v_{kql}\}$ – матрица производительностей труда для k -го ресурса;
- v_{kql} – производительности труда при выполнении k -м ресурсом l -го типа работы для q -го типа требований, где $q \in Q, k \in K, l \in L$.

Если k -й ресурс не умеет выполнять l -го типа работы для q -го типа требований, тогда: $v_{kql} = 0$. Если k -й ресурс умеет выполнять l -й тип работы для q -го типа требований, тогда функция производительности труда является

сложной функцией от множества переменных: $v_{kql} = v(k, q, l, z, \dots)$, где z – размер требования.

Доступные промежутки времени для выполнения работ у k -го ресурса:

- T_k – множество начальных моментов рабочих промежутков времени;
- D_k – множество длительностей рабочих промежутков времени.

Календарный план проекта. Календарный план проекта представляет собой план перевода требований из начальных состояний в конечные состояния с фиксацией конкретных значений для всех вероятностных величин и неопределенностей.

Основой для построения календарного плана является длительность выполнения каждой работы по переводу требований из одного состояния в другое. Длительность перевода r -го требования из i -го состояния в j -е состояние можно задать в виде следующей функции:

$$d_{rij} = \frac{z_r}{v(k, q, w_{ij}, z_r, \dots)}, \quad (1)$$

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

z_r – размер требования r -го требования, $r \in R$,

k – номер ресурса, который осуществляет данную работу;

q – тип требования;

w_{ij} – тип работы для перевода i -го состояния в j -е состояние.

Календарный план должен также включать оценку последовательности изменения состояний требований с учетом матрицы вероятностей перехода P .

Методика построения модели динамики развития проекта. Решение данной задачи должно включать в себя:

- оценку параметров проекта: длительность, трудозатраты, стоимость ресурсов, уровень качества;
- оценку потребностей в ресурсах и их загрузку во временном разрезе;
- оценку календарного плана проекта;
- оценка динамики изменения состояния готовности требований и размера требований в каждом из состояний.

Описанная ранее формальная модель структуры проекта может быть использована для решения перечисленных выше задач. Однако для этого необходимо еще дополнить модель следующими возможностями:

- выбрать методы и алгоритмы для проведения моделирования;
- разработать метод оценки производительности труда ресурсов, т.е. метод построения функции $v_{kql} = v(k, q, l, z, \dots)$;
- разработать функцию динамического изменения множества требований R с течением времени $R = R(t)$.

Оценку каждого параметра лучше всего выдавать в виде ряда значений с указанием уровня достоверности каждого значения, т.е. вероятности достижения такого значения. На рис. 3 показан график оценки даты окончания проекта, которое зависит от вероятной длительности проекта. Аналогичные оценки должны выдаваться и для трудозатрат, стоимости, уровня качества и прочих параметров.

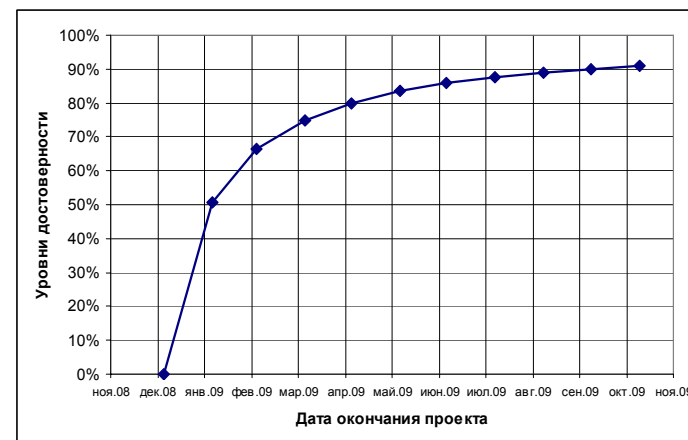


Рис. 3. Пример оценки даты окончания проекта с учетом уровня достоверности оценки

Решение задачи моделирования динамики развития проекта позволит также выдавать оценку динамики изменения состояния требований и размера требований в каждом из состояний в виде, представленном на рис. 4.

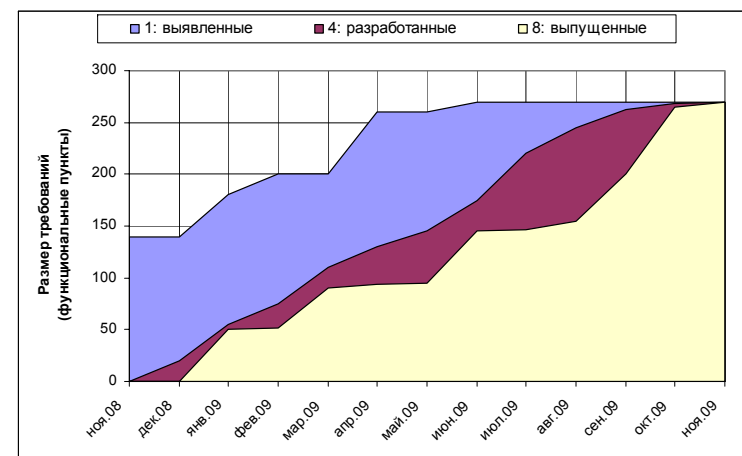


Рис. 4. Динамика изменения размера требований в разных состояниях готовности

Выводы. В данной статье показан метод формального описания модели структуры проекта по разработке ПО. Кроме того, была сформулирована задача прогнозирования динамики развития проекта, для решений которой предполагается использовать разработанную модель структуры проекта. Описанная модель включает в себя модель изменения состояний требований к программному продукту. С математической точки зрения мы получили модель, которая обладает несколькими важными свойствами:

- требования к программному продукту являются объектами, которые могут иметь несколько состояний;
- состояние объектов меняется по заданному алгоритму от порождающего состояния к поглощающему состоянию;
- переход из одного состояния в другое задается некоторой вероятностью перехода;
- объект всегда находится в состоянии некоторое время;
- с течением времени могут появляться новые требования, т.е. могут порождаться новые объекты;
- процесс является конечным.

Перечисленные свойства позволяют сказать, что данный процесс может быть описан моделью марковских процессов. В дальнейшем предполагается применить методы моделирования марковских процессов для решения поставленной задачи моделирования динамики развития проекта.

Список литературы: 1. Макконнелл С. Сколько стоит программный проект (Software Estimation: Demystifying the Black Art). – Издательство «Питер», Русская Редакция, 2007. – 304 с. 2. 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. 3. Lakey P. A Hybrid Software Process Simulation Model for Project Management // Proceedings of ProSIM 2004 Conference, prosim.pdx.edu – 2004. 4. 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. 5. 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. 6. 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. 7. Первозванский А.А. Математические модели в управлении производством. – Главная редакция физико-математической литературы издательства «Наука», М., 1975. – 616 с. 8. Соммервилл И. Инженерия программного обеспечения, 6-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2002. – 624 с. 9. A Guide to the Project Management Body of Knowledge (PMBOK® Guide), 3rd edition. – Project Management Institute, Pennsylvania, USA, 2004. – 401 p.

Поступила в редколлегию 15.12.08