УДК 651. 34

Г.Г. Швачич, С.Г. Семенов, М.И. Главчев, Кассем Халифе

Национальный технический университет «ХПИ», Харьков

МОДЕЛЬ РАСЧЕТА ВРЕМЕННЫХ ГРАНИЦ ПРОЕКТОВ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В статье обозначена необходимость прогнозирования временных затрат на разработку программного обеспечения (ПО) и представлена обобщенная математическая модель для расчета временных границ проектов. С целью прогнозирования разработан комплекс математических моделей основных этапов разработки программного обеспечения. Разработана математическая модель этапа инициализации процесса разработки ПО, основанная на концептуальных положениях Agile, что позволило выделить ряд наиболее важных параметров оценки временных затрат инициализации и определить их зависимости от качественных характеристик участников проекта. Усовершенствована математическая модель этапа реализации функционала ПО, отличающаяся от известных учетом показателей безопасного программирования.

Ключевые слова: *безопасное программирование, временные затраты на разработку ПО, SCRUM, Agile.*

Введение

Постановка проблемы. Проведенные исследования показали, что в настоящее время существует несколько наиболее популярных методологий разработки ПО. Среди них целесообразно выделить семейство «гибких» методологий Agile (XP, SCRUM) и методологии «бережного производства» (Kanban). Следует заметить, что в литературе [1, 4-7] и такие методологии зачастую относят к разряду «гибких». У каждой из них есть свои особенности, которые стоит учитывать, выбирая ту или иную методологию для управления проектом.

Кроме того для успешного управления процессом разработки программного обеспечения бывает недостаточно выбрать ту или иную методологию и следовать ей на протяжении всего процесса. В случае достаточно большого проекта, разработка которого ведется достаточно долгий период времени, важной может оказаться способность быстро адаптировать используемую в данный момент методологию в соответствии с изменяющимися обстоятельствами, то есть по существу синтезировать различные варианты использования «гибких» и «бережных» методологий в один проект.

Таким образом, в настоящее время вопросы, связанные с оптимизацией процесса разработки ПО, несмотря на многообразие «гибких» методологий управления, остаются актуальными.

Анализ литературы [4-7] показал, что одним из первых этапов процесса разработки ПО является этап планирования. На данном этапе команда решает, как она будет достигать цели, поставленной на предыдущем этапе. Этот этап часто разделяется на два этапа – верхнеуровневое планирование и детальное планирование. На верхнем уровне определяются общие моменты исполнения проекта. Именно на этом этапе должны выполняться задачи прогнозирования временных затрат на выполнение работ. Результатами данного прогноза должны стать данные о сроках выполнения (окончания) различных этапов (заданий) в рамках отдельного проекта (время необходимое на разработку и уточнение документации, кодирование, тестирование, верификации и др.). Затем проводится детальное планирование, на котором составляются финальные планы реализации проекта, и проводится корректировка стратегического плана в зависимости от возможно изменяющихся обстоятельств.

Проведенные исследования, а так же анализ литературы [2, 3] показали, что в настоящее время на практике для решения этих задач практически не используются современные методы интерполяции и экстраполяции. Существующие методики расчёта предлагают только грубые оценки, а необходимые для выполнения задач ресурсы определяются исключительно на основе опыта и субъективного мнения людей, выступающих в роли экспертов, далеко не всегда являющихся специалистами в данной области. Кроме того при описании методологии «бережного производства» (Kanban) это зачастую приводит к неудовлетворительной точности полученных результатов оценки сроков выполнения (окончания) различных этапов в рамках проекта разработки ПО.

Одним из путей решения поставленной задачи прогнозирования является использование подхода, основанного на оценке временных затрат на отдельные этапы разработки ПО, с учетом функций зависимости текущего числа активных дефектов приложения от времени, полученных экспериментальным путём а так же с помощью математического моделирования. Такое комплексное использование априорных и апостериорных данных должно позволить учесть специфику современных методик разработки

ПО с возможным динамическим изменением (расширением) рамок проекта по желанию заказчика либо по иным причинам.

Модель для расчета временных границ проектов разработки ПО

Анализ перечисленных выше методологий разработки программного обеспечения показал, что в практически каждом проекте можно выделить три обязательных этапа: инициация, реализация функционала и тестирование. Работы, выполняемые на первых двух этапах, структурированы и представляют собой совокупность действий «мозговых штурмов» A1, A2 и т.д., на которых определяется что же должен представлять из себя продукт проекта и реализацию функционала B (реализация подзадачи B1 и т.д.).

Для математической формализации первого этапа разработки ПО воспользуемся следующими допущениями.

Пусть $A = \bigcup_{i=1}^{N} A_i$ – множество характеристик вопросов, вынесенных на рассмотрение в процессе «митинга», где N – количество вопросов, которые рассматриваются. $A_i = \langle a_i, b_i, n_i \rangle$ – кортеж общих характеристик, где a_i – время, отведенное докладчику i – го вопроса, b_i – время, отведенное на обсуждение i – го вопроса, n_i – количество участников «митинга», участвующих в обсуждении i -го вопроса.

Пусть $\tilde{a}_i, \tilde{b}_i$ – случайные величины, имеющие распределения Q_{a_i}, Q_{b_i} соответственно. При этом Q_{a_i} имеет усеченное экспоненциальное распределение с матожиданием $m[\tilde{a}_i] = a_i + \Delta t_i$, где Δt_i – отклонение времени выступления докладчика при обсуждении i – го вопроса. (обобщенный пример-иллюстрация усеченного экспоненциального распределения представлен на рис. 1). Также следует заметить, что характеристика Q_{b_i} это сумма распределений $Q_{b_{ij}}$, где j – номер выступления при обсуждении вопроса i. Учтем, что $n_i > \tilde{n}_i$ – количественный состав группы разработки ПО, принявших участие в обсуждении вопросов.

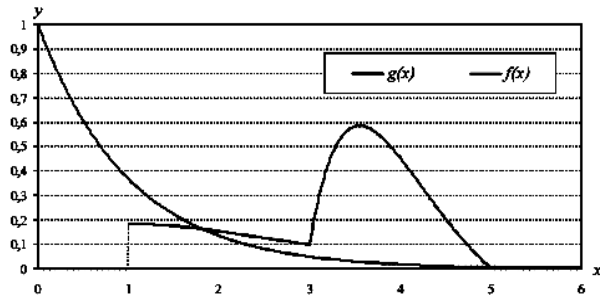


Рис. 1. Пример-иллюстрация усеченного экспоненциального распределения

Проведенные исследования показали, что распределение $Q_{b_{ij}}$ можно аппроксимировать усеченным экспоненциальным распределением с математическим ожиданием $m[\tilde{b}_{ij}] = b_i/\tilde{n}_i + \Delta t_{ij}$, где Δt_{ij} – отклонение времени обсуждения i -го вопроса j -м участником обсуждения. Следует заметить, что при увеличении $\tilde{n}_i$ или Δt_{ij} распределение Q_{b_i} приближается к усеченному нормальному (обобщенный пример-иллюстрация усеченного нормального распределения представлен на рис. 2). Учитывая линейные особенности математического ожидания, можно рассчитать математическое ожидание общих характеристик времени, необходимого для проведения «митингов» без определения его распределения.

$$M[T] = \sum_{i=1}^{N}\left(a_i + \Delta t_i + \sum_{j=1}^{\tilde{n}i}\left(\frac{b_i}{\tilde{n}_i} + \Delta t_{ij}\right)\right). \qquad (1)$$

Используем предложенные математические допущения для предварительных расчётов на этапах инициации, реализации функционала и разработаем комплекс математических моделей описывающих указанные этапы разработки ПО.

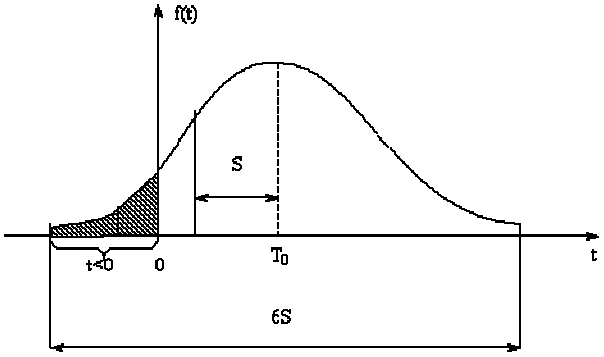


Рис. 2. Пример-иллюстрация усеченного нормального распределения

2. Комплекс математических моделей этапов инициации, реализации функционала ПО

2.1. Математическая модель этапа инициализации процесса разработки ПО

Из литературы [4 – 7], описывающей процесс управления разработкой ПО известно, что в настоящее время в гибких методологиях используются ряд положений, суть которых зафиксирована в манифесте Agile, и которые кратко можно сформулировать следующим образом:

– разработка ведется короткими циклами (итерациями), продолжительностью 1-4 недели;

– в конце каждой итерации заказчик получает ценное для него приложение (или его часть), которое можно использовать в бизнесе;

– команда разработки сотрудничает с Заказчиком в ходе всего проекта;

– изменения в проекте приветствуются и быстро включаются в работу.

Анализ примеров практической реализации гибких методологий позволил определить, что этап инициализации процесса разработки ПО в соответствии представленными положениями включает в себя ряд мероприятий (митингов), имеющих различные тактические и стратегические цели. Например, Daily meetings (Ежедневный контроль). Daily meetings, иначе называемый Stand-up Meeting проводится каждый день. На этом мероприятии каждый член команды должен отчитаться о проделанной работе в течении прошлого дня, наметить план работы на день и решить, существующие к остальным участникам «митинга», вопросы. Продолжительность такого мероприятия должна быть не более 15 минут. При этом за регламент «митинга» следит Scrum-мастер.

С точки зрения временной оптимизации этого процесса существенных «выигрышей» добиться сложно, поскольку данное мероприятие не занимает существенного рабочего времени.

В то же время еще одним подобным мероприятием в гибких методологиях является «retrospective meeting» (Ретроспективное совещание). Именно эти мероприятия занимают значительное время, и могут быть неэффективными при несоблюдении регламента, методологий проведения мероприятия, низкого профессионального уровня участников и других факторов. Рассмотрим более подробно данный процесс и математически формализуем его.

Для математической формализации первого этапа предлагается воспользоваться 4 параметрами оценки временных затрат: длительностью первоначальной коллективной оценки сложности проекта $T_{оц}$, временными затратами на отчет о проделанной работе с момента предыдущего SCRUM-митинга $T_{от}$, временными затратами на обсуждение проблем, возникших во время работы $T_{об}$, время на постановку задач до следующего митинга $T_{пост}$.

Тогда время инициации $T_{и}$ можно описать как:

$$T_{и} = T_{оц} + T_{от} + T_{об} + T_{пост}. \qquad (2)$$

При этом указанные временные затраты зависят от ряда объективных и субъективных факторов. В целом, используя математические допущения и выражение (1), эти показатели можно описать следующим образом.

Одним из наиболее важных и детально описанных процессов в Scrum, является процесс планирования «спринта», промежутка времени в течении которого выполняется работа над продуктом. Длительность первоначальной коллективной оценки сложности проекта $T_{оц}$ зависит во многом от методики оценки, числа участников SCRUM-митинга, их профессиональной подготовки и коммуникативности, что существенно влияет на количество разногласий и спорных моментов при оценке сложности проекта.

Рассмотрим одну из наиболее популярных методик планирования и оценки сложности проекта – покер планирования (англ. Planning Poker, а также англ. Scrum poker). Это методика оценки проектов при разработке программного обеспечения, главной целью которой является достижение договорённости, относительно сложности предстоящей работы или объёма решаемых задач [6].

Анализ данной методики показал, что среднее время первоначальной коллективной оценки сложности проекта можно описать как

$$T_{оц} = \sum_{g=1}^{n_{итт_g}} \left(\frac{t_{итт_g}}{n_{итт_g}} + \Delta t_g + \sum_{j=1}^{\hat{n}_g} \left(a_{g_j} + \Delta t_j \right) \right), \qquad (3)$$

где $t_{итт_g}$ – время, необходимое на проведение g-й итерации оценки сложности проекта; $n_{итт_g}$ – число итераций, необходимое для достижения консенсуса; Δt_g – отклонение времени проведения g-й итерации оценки сложности проекта; a_{g_j} – время, отведенное участникам с высокими и низкими оценками сложности проекта на g-й итерации; Δt_j – отклонение времени высказывания участников обсуждения в g-й итерации при обосновании своей оценки; $\hat{n}_g$ – число участников с высокими и низкими оценками сложности проекта на g-й итерации.

Следует заметить, что время Δt_g чаще всего не велико и существенно не влияет на общее время оценки сложности проекта. Связано это с тем, что это показатель во многом зависит от квалификации только одного участника – модератора, который следит за временем проведения итерации. Однако уровень профессиональной подготовки и коммуникативности остальных участников SCRUM-митинга существенно влияет на показатель Δt_{ij}, который в свою очередь может изменяться в большом диапазоне. Исходя из этого, данный показатель определим следующим образом.

$$\Delta t_j = v \cdot e^{-k \cdot c \cdot \hat{n}_g}, \qquad (4)$$

где k – коэффициент, характеризующий средний уровень профессиональной подготовки участников SCRUM-митинга (варьируется от 0,1 до 0,3); c – коэффициент, характеризующий средний уровень коммуникативности участников SCRUM-митинга (варьируется от 1 до 3); v – усредненный коэффициент сложности решаемой обобщенной задачи.

Для учета временных затрат на отчет о проделанной работе с момента предыдущего SCRUM-митинга $T_{от}$ воспользуемся мнениями экспертов, которые определяют, что чаще всего этот показатель ограничивается регламентом, устанавливаемым

модератором. Поэтому в математической модели этапа инициализации процесса разработки ПО его можно принять за константу, при этом используя практические данные фирм-разработчиков ПО.

Проведенные исследования показали, что для определения затрат на обсуждение проблем, возникших во время работы $T_{об}$ можно воспользоваться положениями, описанными выше (выражение (1)). Тогда среднее время на обсуждение проблем, возникших во время работы, формализуем следующим выражением.

$$T_{об} = \sum_{i=1}^{\breve{N}} \left(a_i + \Delta t_i + \sum_{j=1}^{\breve{n}_i} \left(\frac{b_i}{\breve{n}_i} + \Delta t_{ij} \right) \right), \qquad (5)$$

где $\breve{N}$ – количество проблем, возникших во время работы, и вынесенных на обсуждение в SCRUM-митинге; $\breve{n}_i$ – количество участников обсуждения озвученных проблем.

Несложно заметить, что сокращение временных затрат на обсуждение проблем напрямую связано с временными показателями Δt_i и Δt_{ij}, которые соответственно зависят от профессиональной подготовки участников команды разработчиков и их числа. Проведенные исследования показали, что эти характеристики можно описать таким образом:

$$\Delta t_i = z1 \times e^{-\left(k/\breve{N}\right)}, \qquad (6)$$

$$\Delta t_{ij} = z2 \times e^{-\left(k \cdot \breve{n}_i\right)}, \qquad (7)$$

где z1 и z2 – усредненные коэффициенты сложности проблем обсуждения и управления участниками обсуждения соответственно.

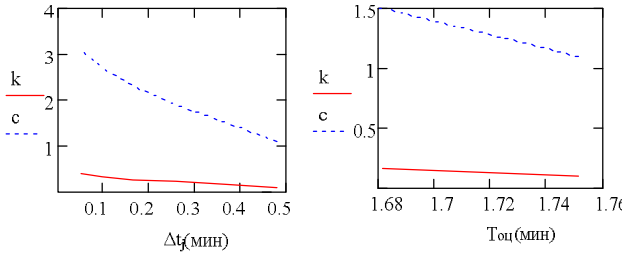
Как указано в начале подраздела еще одним показателем, входящим в общее аналитическое выражение для расчета среднего времени инициализации процесса разработки ПО, является время на постановку задач до следующего митинга $T_{пост}$. Для расчета данного показателя также как и в предыдущем случае воспользуемся выражением 1. Тогда время $T_{пост}$ равно:

$$T_{пост} = \sum_{i=1}^{\breve{\breve{N}}} \left(\breve{\breve{a}}_i + \breve{\Delta t}_i \right), \qquad (8)$$

где $\breve{\breve{N}}$, $\breve{\breve{a}}_i$ – количество задач и время, отведенное на поставку i-й задачи до следующего SCRUM-митинга соответственно, $\breve{\Delta t}_i = \exp\left(-\breve{\breve{N}}/r\right)$ – отклонение времени на поставку i-й задачи до следующего SCRUM-митинга, r – усредненный коэффициент сложности задач ($r = \{2,3,\ldots 45\}$).

Проведем исследования степени влияния, приведенных в подразделе характеристик, на общее время проведения SCRUM-митинга. На рис. 3, а приведены кривые графиков зависимости отклонения Δt_{ij} времени высказывания участников обсуждения в g-й итерации при обосновании своей оценки от коэффициента k, характеризующего средний уровень профессиональной подготовки участников SCRUM-митинга и коэффициента c, характеризующего средний уровень коммуникативности участников SCRUM-митинга, в условиях когда $v = 0,6$, $\hat{n}_g = 2$. Как видно из этих графиков коэффициенты существенно (до 2,5 раз) увеличивают время Δt_{ij}.

На рис. 3, б приведены кривые графиков зависимости длительности $T_{оц}$ первоначальной коллективной оценки сложности проекта от от коэффициента k, характеризующего средний уровень профессиональной подготовки участников SCRUM-митинга и коэффициента c, характеризующего средний уровень коммуникативности участников SCRUM-митинга, в условиях когда $v = 0,6$, $\hat{n}_g = 2$, $t_{итт_g} = \{0,5\ldots 1\}$ мин., $n_{итт_g} = 6$, $\Delta t_g = 0,1$ мин., $a_{g_j} = \{0,1\ldots 0,22\}$ мин.

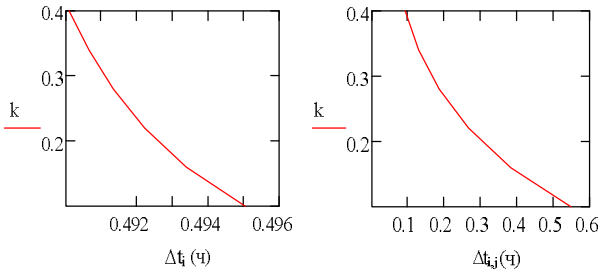


а б

Рис. 3. Графики зависимости отклонения Δt_{ij} от коэффициентов k и c

Как видно из приведенных на рис. 3.3.б графиков улучшение показателя среднего уровня коммуникативности участников SCRUM-митинга в 1,1 раза (до 10 минут в одном часе) уменьшит время первоначальной коллективной оценки сложности проекта. Следует заметить, что приблизительно аналогичных результатов можно добиться при повышении уровня профессиональной подготовки участников SCRUM-митинга.

Проведем исследования модели процесса обсуждения проблем, возникших во время работы. На графиках рис. 4 приведены кривые зависимости временных показателей Δt_i и Δt_{ij} от коэффициента, характеризующий средний уровень профессиональной подготовки участников SCRUM-митинга, в условиях, когда $a_i = \{20,30,\ldots,70\}$ с, $b_i = \{10,20,\ldots,60\}$ с, $\breve{n}_i = 6$. Как видно из этих графиков повышение коэффициент k профессиональной подготовки в 4 раза приблизительно в 1,013 раз (приблизительно 36 с.) уменьшает время Δt_i, и в 6 раз уменьшает время Δt_{ij} (приблизительно 3,5 минуты).



а б

Рис. 4. Графики зависимости временных показателей Δt_i и Δt_{ij} от коэффициента k

Заметим, что в целом повышение уровня профессиональной подготовки участников проекта уменьшает время, необходимое на обсуждение проблем, до 4 минут (в зависимости от входных данных), что наглядно иллюстрирует график рис. 5.

Так на рис. 5 приведена кривая графика зависимости времени обсуждения проблем, возникших во время работы $T_{об}$ от коэффициента k в условиях, когда $\sum_{i=1}^{\breve{N}}(a_i) = 2$ мин., и $\sum_{i=1}^{\breve{N}} \frac{b_i}{\breve{n}_i} = 1$ мин.

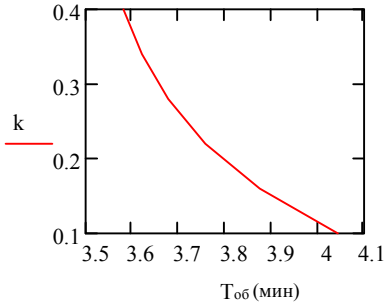


Рис. 5. График зависимости $T_{об}$ от коэффициента k

Исследуем взаимовлияние показателей, используемых в аналитическом выражении для расчета времени на постановку задач до следующего митинга $T_{пост}$. На рис. 6 приведен график зависимости отклонения времени $\Delta\overset{=}{t}_i$ на поставку i-й задачи до следующего SCRUM-митинга от усредненного коэффициента сложности задач r, выполненный при условии, что $\overset{=}{N}$ – количество задач до следующего SCRUM-митинга равно 6.

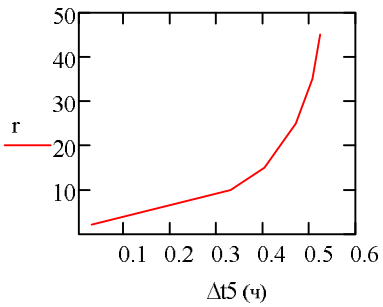


Рис. 6. Зависимость отклонения времени $\Delta\overset{=}{t}_i$ от усредненного коэффициента сложности задач r

Данный график наглядно иллюстрирует, что увеличение усредненного коэффициента сложности задач в 4 раза приводит к увеличению отклонения времени в 1,5 раз. Следует заметить, что в соответствии с выражением 3.8. можно наблюдать аналогичную зависимость времени на постановку задач до следующего митинга $T_{пост}$ от коэффициента сложности задач.

Оценим степень влияния коэффициентов k и c на общее время SCRUM-митинга. На рис. 7 представлен график зависимости времени инициации разработки ПО $T_и$ от показателей k и c в условиях когда временные затраты на отчет о проделанной работе с момента предыдущего SCRUM-митинга $T_{от}$ =50мин, $a_i = \{20,30,\ldots,70\}$ c, $b_i = \{10,20,\ldots,60\}$ c, $\breve{n}_i = 6$, $\overset{=}{N}$ =6, $v = 0,6$, $\hat{n}_g = 2$.

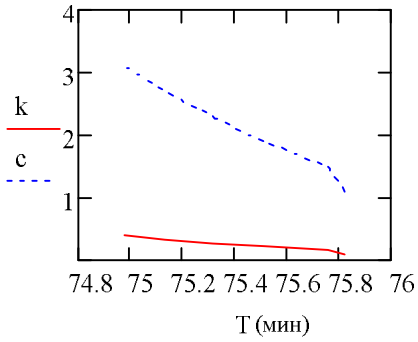


Рис. 7. График зависимости времени $T_и$ от показателей k и c

Как видно из этого графика повышение коэффициентов k и c в указанных выше размерах уменьшает время проведения этапа инициализации процесса разработки ПО (по графику на минуту).

Следует заметить, что представленные результаты имеют больше качественный характер. Это вызвано тем, что величины Δt_j, $\Delta\overset{=}{t}_i$, Δt_i, а так же $\Delta t_{i,j}$ могут иметь как положительное, так и отрицательное значение. Кроме того выбранные эмпирическим путем значения коэффициентов имеют скорее качественный чем количественный характер, и могут варьироваться в зависимости от определенной экспертной оценки. Кроме того, при моделировании не учитывалось, что в командах в настоящее время не задействуются специалисты безопасного программирования и тестирования безопасности. А это в свою очередь существенно снижает средний уровень подготовки участников команды, и соответственно коэффициент k может быть существенно ниже (меньше 0,01) указанного в работе диапазона значений.

Следующим этапом после инициализации является этап реализации функционала ПО. Разработаем модель, формализующую данный этап.

2.2. Математическая модель этапа реализации функционала ПО

Проведенные анализ литературы [3-7] и исследования показали, что в настоящее время существует ряд подходов к математической формализации этапа реализации функционала ПО. В абсолютном большинстве эти модели подразумевают простейший вариант – линейную оценку времени, необходимого для реализации проекта. С одной стороны это упрощает модель, а с другой стороны дает возможность внесения изменений (уточнений) в случае возникновения дополнительных факторов, влияющих на точность результатов моделирования.

В качестве основы воспользуемся подходом, описанным в работе [3], где модель оценки временных затрат на реализацию функционала ПО предлагается ограничить тремя параметрами: сложностью выполнения, важностью точности вычислений и новизной. Однако, как было указано в предыдущем разделе пренебрежение фактором безопасности ПО в значительной степени ухудшает его качество. Поэтому, большинство фирм-разработчиков ПО для учета фактора безопасности выделяют дополнительные ресурсы и силы. Исходя из этого усовершенствуем математическую модель этапа реализации функционала ПО, путем учета фактора безопасного кодирования ПО. В этом случае время необходимое на реализацию функционала ПО можно представить в виде выражения:

$$T_{реал} = K_{рез} \times \times \sum_{\ell} \left(K_{слож} D + K_{важн} П_{важн} + K_{нов} П_{нов} + K_{без} G \right), \quad (9)$$

где $K_{рез}$ – коэффициент резервного времени, связанный с учетом различных рисков проекта, среднего времени задержки выполнения задач в команде. В идеальном случае он может быть принят за 1, но на практике изменяется в пределах от 1,3 до 1,5; $K_{слож}$ – коэффициент сложности задачи; D – сложность задачи, выраженная в человеко-часах; $K_{важн}$; $П_{важн}$ – коэффициент и параметр важности точности вычислений. В любых программах, в частности, ориентированных на финансовые операции, чрезвычайную важность приобретает точность вычислений. Известен случай, когда из-за возможной ошибки в одном из младших разрядов при вычислениях с плавающей точкой компания Intel проводила кампанию по отзыву своих процессоров; $K_{нов} П_{нов}$ – коэффициент и параметр новизны решения; $K_{без} G$ – коэффициент и параметр безопасности ПО. Коэффициент безопасности определяется уполномоченной организацией, проводящей тестирование ПО на безопасность, а параметр безопасности определяется временем на реализацию функциональности, соответствующим требованиям грифа секретности ПО; ℓ – число требований к системе.

Следует заметить, что число факторов и степень их влияния на проект не являются жестко заданными, однако для конкретного проекта могут быть выведены аналитически. Данные факторы можно как априорно так и апостериорно закладывать в параметры сложности задачи или новизны решения. Например, время добавления в проект дополнительных идентичных SQL-запросов линейно зависит от числа запросов (повышается сложность), а необходимость изучения ранее не применявшейся технологии в ходе выполнения проекта (в том числе и правил безопасного кодирования) может привести к экспоненциальному росту времени выполнения поставленной задачи.

Оценим показатель времени необходимого на реализацию функционала ПО в соответствии с выражением (9). На рис. 8 представлен график зависимости времени реализации функционала ПО $T_{реал}$ от коэффициента резервного времени $K_{рез}$, полученный при условии, что $K_{слож}$ варьируется от 2 до 45, D – сложность задачи равна $D = \{40, 45, \ldots, 65\}$ чел-час, $K_{важн} = \{1, 2, \ldots 6\}$, $П_{важн} = \{2, 4, \ldots, 12\}$ чел-час, $K_{нов} = \{1, 2, \ldots, 6\}$, $П_{нов} = \{2, 3, \ldots, 7\}$ чел-час, $K_{без} = \{1, 2, \ldots 6\}$, $G = \{6, 8, \ldots, 16\}$ чел-час.

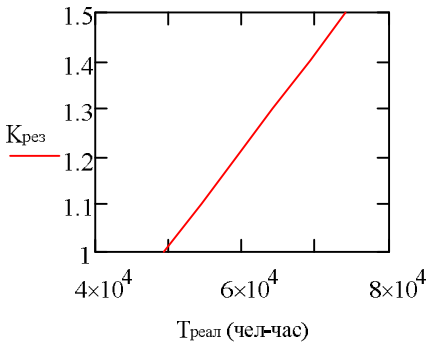


Рис. 8. График зависимости времени реализации функционала ПО $T_{реал}$ от коэффициента резервного времени $K_{рез}$

Как видно из графика зависимость времени реализации функционала ПО $T_{реал}$ от коэффициента резервного времени $K_{рез}$ достаточно велика. Так, при $K_{рез} = 1$, $T_{реал} = 4,944 \times 10^4$ чел-час. А при $K_{рез} = 1,3$, $T_{реал} = 6,427 \times 10^4$ чел-час, что в 1,3 раз больше предыдущего результата. В целом, полученные результаты времени реализации функционала ПО представлены в табл. 1. Следует заметить, что в этой же таблице (столбец 3) приведены значения времени необходимого на реализацию функционала ПО без учета показателей безопасного программирования ($K_{рез}$, G) – $T^*_{реал}$.

Таблица 1

Сравнительные результаты
времени реализации функционала ПО

$K_{рез}$	$T_{реал}$	$T^{*}_{реал}$
1	2	3
1	4.944*104	4.784*104
1.1	5.438*104	5.263*104
1.2	5.933*104	5.741*104
1.3	6.427*104	6.22*104
1.4	6.922*104	6.698*104
1.5	7.416*104	7.177*104

Как видно из этой таблицы, пренебрежение учетом показателей безопасного программирования занижает прогнозируемое время в 1,033 раза, и это с учетом того, что для входных данных показателей безопасного программирования ($K_{без}$, G) взяты достаточно низкие значения. На практике эти значения могут быть больше и, соответственно, прогнозируемое время так же увеличивается.

Выводы

Разработана математическая модель этапа инициализации процесса разработки ПО, основанная на концептуальных положениях Agile, что позволило выделить ряд наиболее важных параметров оценки временных затрат инициализации и определить их зависимости от качественных характеристик участников проекта.

Усовершенствована математическая модель этапа реализации функционала ПО, отличающаяся от известных учетом показателей безопасного программирования. Это позволило повысить точность результатов моделирования на 3%.

Дальнейшее развитие данные модели получат в способе масштабирования существующей методологии разработки с учетом требований безопасности программного обеспечения.

Это позволит повысить безопасность проекта и обеспечивать как быстрый рост функционала, так и приемлемый уровень качества сервиса.

В комплексе синтез разработанных математических моделей и способа масштабирования позволили усовершенствовать метод масштабирования методологии разработки программного обеспечения с учетом требований безопасности, отличающийся от известных возможностью управления существующими в организации (фирме) силами (специалистами) как в составе команды, так и в плоскости специалистов смежного направления (специалистов безопасного программирования и тестирования безопасности ПО).

Список литературы

1. Вэйдер Майкл Томас Инструменты бережливого производства. Мини-руководство по внедрению методик бережливого производства / Майкл Томас Вэйдер – Альпина Паблишер, 2012, 125 с.

2. Гмурман В.Е. Теория вероятностей и математическая статистика / В.Е. Гмурман. – М.: Высшая школа, 2003. – 479 с.

3. Полицын С. А. Подходы к вычислению временных затрат на проекты в сфере разработки программного обеспечения на основе использования прецедентов / С.А. Полицын // Программная инженерия №7 2011 С.9-14

4. Канер Сем Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений / С.Канер. – К.:ДиаСофт, 2001. – 544 с.

5. Макконнелл С. Сколько стоит программный проект / С. Макконнелл – Питер, 2007 – 304 с.

6. Kniberg Henrik Scrum and XP from the Trenches - 2nd Edition / Henrik Kniberg – InfoQ 2015 – 94 с.

7. Ruby Sam Agile Web Development with Rails / Sam Ruby, Dave Thomas, David Heinemeier Hansson – The Pragmatic Bookshelf Год: 2011, 2011 – 480 с.

Надійшла до редколегії 1.02.2017

Рецензент: д-р техн. наук, проф. О.О. Можаєв, Національний технічний університет «ХПІ», Харків.

МОДЕЛЬ РОЗРАХУНКУ ЧАСОВИХ КОРДОНІВ ПРОЕКТІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Г.Г. Швачич, С.Г. Семенов, М.І. Главчев, Кассем Халіфі

У статті вказано на необхідність прогнозування часових витрат на розробку програмного забезпечення (ПО) і представлена узагальнена математична модель для розрахунку часових меж проектів. З метою прогнозування розроблений комплекс математичних моделей основних етапів розробки програмного забезпечення. Розроблено математичну модель етапу ініціалізації процесу розробки ПО, заснована на концептуальних положеннях Agile, що дозволило виділити ряд найбільш важливих параметрів оцінки тимчасових витрат ініціалізації і визначити їх залежності від якісних характеристик учасників проекту. Удосконалено математичну модель етапу реалізації функціоналу ПЗ, що відрізняється від відомих урахуванням показників безпечного програмування.

Ключові слова: *безпечне програмування, тимчасові витрати на розробку ПО, SCRUM, Agile.*

MODEL OF CALCULATING THE ADVANCED BORDERS OF PROJECTS OF SOFTWARE PROCESSING

G.G. Shvachich, S.G. Semenov, M.I. Glavchev, Kassem Khalifi

The article identifies the need to forecast the time costs for the development of software (software) and presents a generalized mathematical model for calculating the time boundaries of projects. For the purpose of forecasting, a complex of mathematical models of the main stages of software development has been developed. A mathematical model of the initialization phase of the software development process was developed, based on the Agile conceptual provisions, which made it possible to identify a number of the most important parameters for estimating the time costs of initialization and to determine their dependence on the qualitative characteristics of the project participants. The mathematical model of the implementation phase of the software functional is improved, which differs from the known ones taking into account the indices of safe programming.

Keywords: *safe programming, time costs for software development, SCRUM, Agile.*