ВЫБОР РАЦИОНАЛЬНОЙ МОДЕЛИ РАЗРАБОТКИ БЕЗОПАСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Жидков И.В.[1], Зубарев И.В.[2], Хабибуллин И.В.[3]

Цель исследования: *обоснование подходов к выбору рациональной модели разработки программного обеспечения на основе предлагаемой системы критериев.*

Метод исследования: *применялись методы теории принятия решений, метод ранжирования систем по совокупности скалярных компонент, в частности метод «жёсткого» ранжирования.*

Результат исследования: *проведен анализ основных моделей разработки программного обеспечения, рассмотрена система критериев, используемая для сравнения различных моделей разработки программного обеспечения, разработаны предложения по выбору рациональной модели разработки программного обеспечения на основе решения задачи гипервекторного ранжирования.*

Ключевые слова: *программное обеспечение, жизненный цикл программного обеспечения, модель разработки, метод ранжирования.*

DOI: 10.21681/2311-3456-2021-5-21-29

1. Введение

Каждая из моделей разработки программного обеспечения (ПО) имеет присущие ей преимущества и недостатки, определяющие ее применение для конкретного проекта. Оптимальный выбор модели жизненного цикла (ЖЦ) ПО является комплексной задачей, требующей продолжительных исследований в данной области, накопления и систематизации знаний. Неверно выбранная модель может привести к перерасходу ресурсов и затягиванию сроков выполнения проекта, вплоть до полного провала [1]. Кроме того, правильный выбор модели ЖЦ ПО положительно скажется на его качестве и безопасности [2-6].

В литературе встречается несколько подходов к сравнению различных моделей разработки ПО.

В одном из подходов ключевыми критериям выбора методологии являются отношение методологии к каскадной или итерационной группе разработки, а также степень формальности разработки рабочих материалов. Примерами использования этих показателей для сравнения методологий можно найти, например, в [7, 8].

В [1] сделан вывод, что каждому проекту своя методология», то есть отмечено, что для каждого конкретного проекта «оптимальной» будет одна какая-то методология и существенную роль в поисках нужной методологии играет определение принципов, по которым ее можно было бы разработать. Полагается, что методологии целесообразно классифицировать по размеру команды разработчиков и критичности системы, так как именно они лучше всего подходят для изначальной оценки.

Другой подход используется в работе Эрика Дж. Брауде «Технология разработки программного обеспечения» [9]. В ней сравниваются процессы разработки ПО по четырем факторам: легкость контроля документации, возможность взаимодействия с заказчиком, поддержание хорошего проектирования и сбор метрических данных, собранных в ходе проекта. Выбор осуществляется только из трех процессов разработки ПО (водопадного, спирального и инкрементального) и сводится к решению вопроса о количестве итераций.

В курсе лекций «Верификация программного обеспечения» Московского инженерно-физического института [10] С.В. Синицыным и Н.Ю. Налютиным сравниваются отдельно четыре типа жизненного цикла (каскадный; V-образный; спиральный и экстремального программирования) по трем критериям: длина цикла; верификация и внесение изменений; интеграция отдельных компонент системы. Далее осуществляется сравнение технологий MSF, RUP и XP по четырем критериям: оптимальная команда; соответствие стандартам; допустимые технологии и инструменты; удобство модификации и сопровождения.

Приведенные подходы представляются слишком грубыми, так как сравнение моделей ЖЦ ПО осуществляется на качественном уровне, при этом не учитывается ряд других факторов. Исключительно качественное сравнение не позволяет выбрать именно оптимальную для данного проекта модель разработки ПО.

Количественное сравнение моделей разработки ПО используется в работе «Управление программными проектами: достижение оптимального качества при минимуме затрат» [11]. Специально для выбора модели ЖЦ разработаны 32 критерия, сгруппированных

1 Жидков Игорь Васильевич, кандидат технических наук, доцент, сотрудник ФГБУ «3 ЦНИИ» Минобороны России, г. Москва, Россия. E-mail: igorzh@bk.ru

2 Зубарев Игорь Витальевич, кандидат технических наук, доцент, начальник департамента АО "НПК "Техмаш", г. Москва, Россия. E-mail: i.v.zubarev@tecmash.ru

3 Хабибуллин Ильфат Винирович, сотрудник Восьмого управления ГШ ВС РФ, г. Москва, Россия. E-mail: rainaira1632@mail.ru

по четырем категориям: характеристики требований, характеристики участников команды разработчиков, характеристики коллектива пользователей, характеристики типа проектов и рисков. Данный подход используется в сертификационных программах менеджмента качества программных проектов Института качества программного обеспечения (Техасский университет, г. Остин, США), а также программе подготовки на получение звания сертифицированного инженера от Американского общества качества. Однако, приведенная процедура выбора оптимальной модели на использовании независимых таблиц критериев для каждой категории, упорядочении их по степени важности для данного проекта, описана не подробно, из-за чего не ясна. В то же время данный набор критериев представляется одновременно недостаточно полным и отчасти избыточным.

В работе В.В. Бахтизина и Л.А. Глуховой [12] уточняется приведенная выше классификация проектов [11], предназначенная для выбора модели жизненного цикла программных средств. В ней для сравнения к имеющимся 32 критериям добавлены еще три: масштабность, стабильность и критичность продукта проекта. Кроме того, описана процедура выбора модели жизненного цикла программных средств на основе сводной таблицы критериев классификации проектов.

В работах Барабанова А.В. и Маркова А.С. [13, 14], посвященных созданию безопасного программного

Таблица 1

Система критериев, используемая для сравнения различных моделей разработки ПО

Критерии и категории, характеризующие модели разработки ПО	**Обозначение**	**Примечание**
Характеристики требований к проекту	K_1	
Могут ли все требования к проекту не задаваться сразу в начале ЖЦ (до начала проектирования)?	$K_{1.1}$	0 — нет; 1 — да.
Могут ли требования изменяться в процессе итерации?	$K_{1.2}$	0 — нет; 1 — возможно; 2-да.
Приветствуется ли изменение требований даже на поздних этапах разработки ради достижения заказчиком конкурентного преимущества?	$K_{1.3}$	0 — нет; 1 — да.
Необходимо ли демонстрировать требования с целью их определения и/или для проверки концепции?	$K_{1.4}$	0 — нет; 1 — да.
Характеристики команды разработчиков	K_2	
Являются ли проблемы предметной области проекта новыми для большинства разработчиков?	$K_{2.1}$	0 — нет; 1 — да.
Являются ли инструменты, используемые в проекте новыми для большинства разработчиков?	$K_{2.2}$	0 — нет; 1 — да.
Является ли команда разработчиков самоорганизующейся?	$K_{2.3}$	0 — нет; 1 — да.
Могут ли меняться роли участников проекта?	$K_{2.4}$	0 — нет; 1 — да.
Характеристики пользователей и заказчика	K_3	
Частота отслеживания заказчиком хода выполнения проекта.	$K_{3.1}$	Задан интервалами значений. 0 — реже 1 раза в 2 месяца; 1- раз в 1-2 месяца; 2- раз в 1 -2 недели; 3- ежедневно / раз в несколько дней.

Критерии и категории, характеризующие модели разработки ПО	Обозначение	Примечание
Вовлечены ли пользователи в фазы жизненного цикла?	$K_{3.2}$	0 — нет; 1 — частично; 2 — на всех этапах.
Характеристики типов проектов и рисков	K_4	
Будет ли проект являться расширением существующей системы?	$K_{4.1}$	0 — нет; 1 — да.
Будет ли финансирование проекта стабильным на всем протяжении жизненного цикла?	$K_{4.2}$	0 — возможно, нет; 1 — да.
Ожидается ли длительная эксплуатация продукта?	$K_{4.3}$	0 — нет; 1 — возможно; 2-да.
Будет ли продукт изменяться (возможно, с применением непредвиденных методов), на этапе сопровождения (стабильность продукта)?	$K_{4.4}$	Задан интервалами значений. 0 — маловероятно; 1 — возможно; 2 — постоянно.
Масштаб разрабатываемого продукта.	$K_{4.5}$	Задан интервалами значений. 0 — нет; 1 — средний; 2 — да.
Критичность разрабатываемого продукта, влияющая на уровень безопасности ПО.	$K_{4.6}$	Задан интервалами значений. 0 — потеря комфорта; 1 — потеря денег; 2 — потеря больших денег и бизнеса; 3 — потеря жизни.
Характеристики процесса разработки	K_5	
Будет ли проектирование и конструирование отдельно каждого свойства системы / пошаговая детализация?	$K_{5.1}$	0 — нет; 1 — возможно; 2 — да.
Будет ли учитываться требования ГОСТов серий 19 и/или 34?	$K_{5.2}$	0 — нет; 1 — да.
Степень формализации разработки рабочих материалов.	$K_{5.3}$	0 — низкая (работающее ПО выше всеобъемлющей документации); 1 — высокая (детальная документация, большое кол-во документов, формальные процедуры рецензирования); 2 — средняя (документация в основном сконцентрирована на конечном продукте).
Необходимо ли как можно быстрее получить первые версии работающей программы?	$K_{5.4}$	0 — нет; 1 — да.

DOI: 10.21681/2311-3456-2021-5-21-29

обеспечения, разработана концептуальная теоретико-множественная модель, включающая в том числе совокупность мер безопасного программирования с учетом требований «ОК».

Основным недостатком рассмотренных концептуальных подходов является недостаточная проработка выбора критериев при наличии ресурсных и временных ограничений. В данной статье сделана попытка комплексирования указанных подходов по критерию рациональности.

2. Система критериев для сравнения моделей разработки ПО

Для выбора оптимальной модели разработки ПО предлагается использовать систему критериев, представленную в таблице 1.

Представленная в таблице 1 система критериев служит базой для сравнения, оценки и окончательного выбора модели разработки ПО. Она основана на подходе, используемом в работе «Управление программными проектами: достижение оптимального качества при минимуме затрат» [9] – 20 критериев предлагается разделить на 5 категорий (векторных компонент).

Первая векторная компонента – *характеристики требований,* классифицирует проекты в зависимости от возможности изменения требований пользователей и заказчика к разрабатываемому продукту. К ней относятся, например, такие частные критерии, как возможность формулирования требований в начале жизненного цикла, возможность изменения требований на протяжении жизненного цикла.

Вторая векторная компонента – *характеристики команды разработчика,* определяет проекты с учетом особенностей коллектива разработчиков и учитывают тот факт, что от этих особенностей зависит успех реализации проекта. В частности, к данной категории относятся такие частные критерии, как новизна инструментальных средств проекта для большинства разработчиков и новизна предметной области.

Третья векторная компонента – *характеристики пользователей и заказчика,* классифицирует проекты в зависимости от возможной степени участия пользователей (заказчиков) в процессе разработки и их взаимосвязи с командой разработчиков на протяжении проекта. Например, к данной категории относятся такие критерии, как степень вовлеченности пользователя в ЖЦ разработки и частота отслеживания заказчиком хода выполнения проекта. Заметим, что один из критериев задан интервалами значений.

Четвертая векторная компонента – *характеристики типов проектов и рисков,* отражает масштаб проекта, особенности эксплуатации, достаточность ресурсов для его исполнения и т.д. [15]. К данной категории относятся: предполагаемая длительность эксплуатации программных средств (ПС), требования к уровню надежности разрабатываемого ПС и другие частные критерии. Причем три критерия заданы *интервалами* значений.

Пятая векторная компонента – *характеристики процесса разработки,* отражает особенности процесса разработки, например, учет требований российских стандартов серий 19.XXX и 34.XXX, степень формализации разработки рабочих материалов и др. То есть, в данной категории приведены частные критерии, являющиеся отличительными особенностями различных моделей разработки ПО.

3. Математическая постановка задачи

С использованием методов теории принятия решений [16, 17] предлагаемая система критериев позволяет проводить оценку множества различных моделей разработки ПО. Для такой оценки широкое применение нашли различные методы ранжирования. Удачно выбранный метод ранжирования позволяет:

- найти множество эффективных вариантов (множество Парето);
- найти множество неэффективных вариантов;
- построить упорядоченное множество эффективных вариантов (кортеж Парето). Для однозначного понимания введем следующие определения [1]:

Определение 1. Многокритериальными называют задачи, в которых векторный критерий представляет собой упорядоченное множество скалярных компонент.

Определение 2. Многовекторными называют задачи, в которых векторный критерий представляет собой упорядоченное множество векторных компонент, а каждая векторная компонента – упорядоченное множество скалярных компонент.

Определение 3. Гипервекторными называют задачи, в которых векторный критерий представляет собой упорядоченное множество многовекторных компонент, каждая многовекторная компонента – упорядоченное множество векторных компонент, а каждая векторная компонента – упорядоченное множество скалярных компонент.

Определение 4. Кортеж Парето – упорядоченное множество эффективных вариантов.

Определение 5. Подкортеж Парето – упорядоченное подмножество эффективных вариантов.

Как показывает анализ системы критериев, используемых для сравнения различных моделей разработки ПО, задача принятия решений сводится в данном случае к задаче гипервекторного ранжирования [1]. Рассмотрим математическую постановку и метод решения такой задачи.

Введём необходимые в дальнейшем обозначения:

$S = \left\{ S_\alpha, \alpha = \overline{1,n} \right\}$ – множество моделей разработки ПО (моделей);

$K\left(S_\alpha\right) = \left\{ K_\varepsilon\left(S_\alpha\right),\ \varepsilon = \overline{1,E} \right\}$ – множество многовекторных компонент $K_\varepsilon\left(S_\alpha\right)$, характеризующих модель разработки ПО (модель) $S_\alpha \in S$;

$K_\varepsilon\left(S_\alpha\right) = \left\{ K_{\varepsilon j}\left(S_\alpha\right),\ j = \overline{1,r_\varepsilon} \right\}$ – множество векторных компонент, характеризующих модель S_α;

$K_{\varepsilon j}\left(S_\alpha\right) = \left\{ K_{\varepsilon ji}\left(S_\alpha\right),\ i = \overline{1,r_{\varepsilon j}} \right\}$ – множество скалярных

компонент, характеризующих модель S_α;

$A=\left\{\alpha_\varepsilon,\ \varepsilon=\overline{1,E}\right\}$ – множество коэффициентов важности многовекторных компонент, причем $\sum_{\mu=1}^{E}\pm_\mu=1$;

$A_\varepsilon=\left\{\alpha_{\varepsilon j},\ j=\overline{1,r_\varepsilon}\right\}$ – множество коэффициентов важности векторных компонент модели S_α, причем $\sum_{j=1}^{r_j}\alpha_{\varepsilon j}=1,\ \varepsilon=\overline{1,E}$;

$A_{\varepsilon j}=\left\{\alpha_{\varepsilon ji},\ i=\overline{1,r_{\varepsilon j}}\right\}$ – множество коэффициентов важности скалярных компонент, причем $\sum_{i=1}^{r_{ij}}\alpha_{\varepsilon ji}=1,\ j=\overline{1,r_\varepsilon},\ \varepsilon=\overline{1,E}$;

S^π – множество эффективных (Парето оптимальных) систем с числом элементов $n^\pi, S^\pi\in S$;
$S_p^0\in S^\pi$ – модели, которые входят в множество эффективных решений, $p\in\left\{\overline{1,n}\right\}$;

$D\left(S_\alpha\right)=\left\{D_i\left(S_\alpha\right),i=\overline{1,M}\right\}$ – множество ограничений, накладываемых на модели, причем должно выполняться условие

$$D_i\left(S_\alpha\right)\le\left\{D_i^0,i=\overline{1,M}\right\}, \quad (1)$$

где D_i^0 – допустимое значение i-го ограничения, M – число ограничений;

$P=\left\{S_{k_1}^0,S_{k_2}^0,\ldots,S_{k_{n^\pi}}^0\right\}$ – упорядоченное множество эффективных моделей (кортеж Парето); элементы кортежа ранжированы в соответствии с решающими правилами так, что выполняется условие

$$S_{k_1}^0\succ S_{k_2}^0\succ\ldots\succ S_{k_i}^0\succ\ldots\succ S_{k_{n^\pi}}^0$$

где «$\succ$» — знак отношения доминирования, $k_i\in\left\{1,2,\ldots,n\right\}$. Длина кортежа равна n^π;

$P_R=\left\{S_{k_1}^0,S_{k_2}^0,\ldots,S_{k_R}^0\right\}$ – упорядоченное множество эффективных моделей (подкортеж Парето). Длина подкортежа равна R.

Допустим, известны множества A, A_ε, $A_{\varepsilon i}$, S, $K_{\varepsilon i}(S_\alpha)$, $D(S_\alpha)$, $(\alpha=\overline{1,n};\ \varepsilon=\overline{1,E},\ j=\overline{1,r_\varepsilon})$, решающие правила. Требуется найти кортеж Парето P, для элементов которого справедливо

$$K\left(S_p^0\right)=\min_{S_\alpha\in S}K\left(S_\alpha\right),S_p^0\in P \quad (2)$$

и выполняется условие (1).

При построении подкортежа Парето P_R заданной мощности R необходимо определить подмножество упорядоченных моделей, для элементов которого справедливо

$$K\left(S_p^0\right)=\min_{S_\alpha\in S}K\left(S_\alpha\right),S_p^0\in P_R \quad (3)$$

и выполняется условие (1).

Допустим, нам известен метод ранжирования систем по совокупности скалярных компонент $K_{\varepsilon j}(S_\alpha)$, $\varepsilon=\overline{1,E}$, $j=\overline{1,r_\varepsilon}$), например, метод «жёсткого» ранжирования. После его применения будут построены частные кортежи Парето, которые позволят однозначно определить расположение модели разработки ПО относительно других моделей по каждой векторной компоненте. Причём выявляются как доминирующие (доминируемые), так и эквивалентные модели. Это позволяет придать всем векторным компонентам некоторые числа, значения которых зависят от расположения моделей: для доминируемых моделей эти значения больше, чем для доминирующих, а для эквивалентных моделей разработки ПО эти значения будут равными. Назовём такие числа *псевдозначениями* (рангами) векторных компонент. Введение таких псевдозначений позволяет вновь применить метод «жёсткого» ранжирования (число обращений к методу будет равно числу многовекторных компонент, т. е. E раз) и построить частные кортежи Парето.

В результате решения задачи получаем расположение моделей по совокупности многовекторных компонент $K_\varepsilon(S_\alpha)$, $\varepsilon=\overline{1,E}$. Это, в свою очередь, позволяет обоснованно ввести псевдозначения многовекторных компонент и вновь обратиться к методу «жёсткого» ранжирования. В итоге и будет построен искомый кортеж Парето.

Таким образом, для решения задачи гипервекторного ранжирования метод «жёсткого» ранжирования применяется $\left[E+1+\sum_{\varepsilon=1}^{E}r_\varepsilon\right]$ раз.

Заметим, что *многокритериальную* задачу ранжирования можно представить как задачу ранжирования систем с двумя уровнями иерархии критериев, *многовекторную* – с тремя уровнями иерархии критериев, а *гипервекторную* – с четырьмя уровнями иерархии критериев. Нетрудно видеть, что предлагаемая идея последовательного ранжирования, введения псевдозначений для векторных и многовекторных компонент позволяет решать задачу ранжирования моделей с любым числом уровней иерархии критериев. В частности, рассмотренный подход применим и для решения задачи гипервекторного ранжирования, если сложная система характеризуется:

- критериями, заданными интервалами значений;
- критериями, заданными в неформализованном (нечётком) виде.

Замечание 1. При значении $E=1$ имеем задачу *многовекторного* ранжирования.

Замечание 2. При значениях $E=1$ и $r_\varepsilon=1$ имеем задачу *многокритериального* ранжирования.

Замечание 3. Метод гипервекторного ранжирования не зависит от принятого решающего правила.

4. Сущность метода жёсткого ранжирования

Рассмотрим теперь основные идеи метода жёсткого ранжирования. Без потери общности изложение будем

проводить для моделей $S_\alpha = \overline{1,n}$, свойства которых задают с помощью критериев $K_j(S_\alpha)$, $j = \overline{1,r}$.

В ходе решения задачи будем анализировать множество упорядоченных пар моделей S_k, $S_l (k = \overline{1,n};\ l = \overline{1,n};\ k \neq l)$. а результат анализа заносить в специальную оценочную матрицу $||C_{kl}||$.

Определение 6. Под оценочной матрицей $||C_{kl}||$ $(k = \overline{1,n};\ l = \overline{1,n};\ k \neq l)$ понимаем такую матрицу, элементы C_{kl} которой позволяют однозначно определить соотношение между k-й и l-й моделями.

Сущность метода заключается в следующем:

1. На основе попарного сравнения моделей S_k, S_l $(k = \overline{1,n};\ l = \overline{1,n};\ k \neq l)$ определяем элементы C_{kl} оценочной матрицы $||C_{kl}||$. Значения элементов C_{kl} подбирают таким образом, чтобы отсечь неэффективные модели. У эквивалентных моделей S_k, S_l все соответствующие критерии равны. Полагаем, $C_{kl} = 1$, $C_{lk} = 1$.

К числу неэффективных моделей отнесем варианты, у которых:

а) все значения критериев l-й модели хуже, чем у k-й модели, тогда полагаем $C_{kl} = N_2 >> 1$;

б) значения $m (m<r)$ критериев l-й модели хуже соответствующих значений критериев k-й модели при равных соответствующих значениях остальных критериев этих моделей; тогда полагаем $C_{kl} = N_3$, $1 << N_3 < N_2$.

Если же для моделей k, l имеем лучшие, худшие и. возможно, равные критерии, то значение C_{kl} определим по методу, изложенному в [17].

Запишем теперь более строго выражения для элементов оценочной матрицы. Обозначим N_{kl}^{+}, N_{kl}^{-}, $N_{kl}^{=}$, — соответственно подмножества номеров лучших, худших и равных критериев для каждой пары вариантов моделей S_k, $S_l (k = \overline{1,n};\ l = \overline{1,n};\ k \neq l)$. Будем осуществлять попарное сравнение моделей на основе анализа критериев $K_j(S_k)$, $K_j(S_l)$, $j = \overline{1,r}$). Для возможных значений подмножеств номеров N_{kl}^{+}, N_{kl}^{-}, $N_{kl}^{=}$ (соответственно лучших, худших и равных критериев) введем следующие значения элементов оценочной матрицы $||C_{kl}||$:

если

$$N_{kl}^{+} \neq \varnothing,\ N_{kl}^{-} \neq \varnothing,\ N_{kl}^{=} = \left\{\overline{1,r}\right\}, \tag{4}$$

то

$$C_{kl} = 1, C_{lk} = 1; \tag{5}$$

если

$$N_{kl}^{+} = \left\{\overline{1,r}\right\},\ N_{kl}^{-} = \varnothing, N_{kl}^{=} = \varnothing\ , \tag{6}$$

то

$$C_{kl} = N_2, C_{lk} = 0,\ N_2 \gg 1; \tag{7}$$

если

$$N_{kl}^{+} = \varnothing, N_{kl}^{-} = \left\{\overline{1,r}\right\},\ N_{kl}^{=} = \varnothing\ , \tag{8}$$

то

$$C_{kl} = 0, C_{lk} = N_2; \tag{9}$$

если

$$N_{kl}^{+} \neq \varnothing, N_{kl}^{-} = \varnothing,\ N_{kl}^{=} \neq \varnothing\ , \tag{10}$$

то

$$C_{kl} = N_3, C_{lk} = 0,\ 1 \ll N_3 < N_2; \tag{11}$$

если

$$N_{kl}^{+} = \varnothing, N_{kl}^{-} \neq \varnothing,\ N_{kl}^{=} \neq \varnothing\ , \tag{12}$$

$$C_{kl} = 0, C_{lk} = N_3; \tag{13}$$

если

$$N_{kl}^{+} \neq \varnothing, N_{kl}^{-} \neq \varnothing,\ \left|N_{kl}^{=}\right| \geq \varnothing\ , \tag{14}$$

то определим C_{kl} в виде [17]

$$C_{kl} = \sum_{j=N_{kl}^{+}} a_j \cdot \left(\sum_{j=N_{kl}^{-}} a_j\right)^{-1},\ C_{lk} = C_{kl}^{-1} \tag{15}$$

Теорема. Если в l-м $\left(l \in \overline{\{1,n\}}\right)$ столбце оценочной матрицы максимальный элемент больше или равен значению N_2, , то l-й вариант модели не принадлежит множеству эффективных решений.

Доказательство. Из условия теоремы следует, что хотя бы для одного из вариантов k $\left(k \in \overline{\{1,n\}}, k \neq l\right)$ выполняется одно из условий (6), (10). Таким образом, вариант l доминируется вариантом k. Значит, согласно определению множества Парето, l-й вариант не может принадлежать множеству эффективных решений.

2. Для формулировки решающих правил (специальных неформальных правил, зависящих от специфики задачи и позволяющих провести ранжирование систем) введем числа: H_l – количество элементов в l-м столбце оценочной матрицы, значение которых больше единицы; M_l –количество элементов в l-м столбце той же матрицы, значение которых меньше единицы; C_{klmax} – максимальное значение элемента в l-м столбце матрицы $||C_{kl}||$.

Физический смысл чисел:

H_l – показывает, сколько моделей из рассматриваемого множества превышают *l-ю;*

M_l – сколько вариантов доминирует *l-я* модель;

C_{klmax} определяет, во сколько раз *l-я* модель «превышается» k-й $\left(k \in \overline{\{1,n\}}, k \neq l\right)$.

Для реализации «жесткого» ранжирования перейдем от одношагового процесса поиска приоритетного расположения моделей к многошаговому процессу.

Общая идея заключается в следующем. На каждом шаге $t (t=1,2,...,n-1)$ выбираем j-ю модель, лучшую с точки зрения предлагаемого ниже решающего правила. Затем ее номер включаем в множество P и в последующем рассмотрении *j-я* система больше не участвует (в матрице $||C_{kl}||$ вычеркиваем j-ю строку и j-й столбец). Указанная процедура позволяет исключить влияние модели S_j на выбор лучшей модели, проводимый уже на шаге $(t+1)$.

При реализации многошагового поиска необходимо сформировать такие решающие правила, которые позволяют определить на каждом шаге $t (t=1,2,...,n-1)$;

- лучшую (доминирующую) модель либо подмножество кандидатов на эквивалентные доминирующие модели;
- эквивалентные модели из числа кандидатов, а также провести ранжирование моделей, не попавших в эквивалентные.

УДК 004.413.2

При формулировке решающих правил вновь используем, но теперь на каждом шаге t, числа $H^{(t)}_l$, $M^{(t)}_l$, $C^{(t)}_{klmax}$, которые имеют оговоренный ранее физический смысл. Ранжирование, как и прежде, осуществляем лишь для эффективных моделей. Все неэффективные варианты запоминаем и из решения исключаем.

5. Разработка методики «жёсткого» ранжирования

Для обоснования методики, по которой должно проводиться ранжирование, определим решающие правила «жёсткого» ранжирования:

1. Ранжирование необходимо проводить среди эффективных моделей по шагам. Число шагов $t \le (n-1)$.

2. На каждом шаге $t(t=1,2,...,n-1)$ нужно:

- найти числа $H^{(t)}_l$, $M^{(t)}_l$, $C^{(t)}_{klmax}$, и определить лучшую модель S_j с минимальным значением $H^{(t)}_j$ и $C_{lj} \ge 1 \forall l \in \overline{\{1,n\}}, l \ne j$;
- номер j занести в множество P;

исключить из оценочной матрицы j-ю строку и j-й столбец.

Если системы с номерами $l_j \in L_{k(t)} = \{l_1, l_2, \ldots, l_j, \ldots, l_{k(t)}\}$ имеют одинаковые минимальные значения $H^{(t)}_{l_j}$, то лучшей является модель S_{l_j} с максимальным значением $M^{(t)}_{l_j} = \max\limits_{l_j \in L_{k(t)}} M^{(t)}_{l_j}$.

3. Если модели с номерами $l_j \in L_{k(t)} = \{l_1, l_2, \ldots, l_j, \ldots, l_{k(t)}\}$ имеют соответственно одинаковые значения $H^{(t)}_{l_j}$, $M^{(t)}_{l_j}$, то из оценочной матрицы надо выделить подматрицу с номерами столбцов и строк $l_j \left(j = \overline{1, k(t)} \right)$ и провести ранжирование ее элементов.

4. Эквивалентные модели $S_{l_j}, l_j \in L_m \subseteq L_k$, следует упорядочить на основе анализа чисел $H^{(1)}_l$, $M^{(1)}_l$, $C^{(1)}_{klmax}$, полученных на первом шаге.

На основе метода «жесткого» ранжирования сформулируем методики многокритериального и гипервекторного ранжирования.

Методика решения задачи многокритериального ранжирования

1. Провести анализ исходной информации. Определить критерии, по которым будут сравниваться модели, коэффициенты важности критериев или группы коэффициентов важности.
   Вычислить элементы оценочной матрицы $||C_{kl}||$. С этой целью использовать выражения (1.4)-(1.15).
2. Определить неэффективные, в паретовском смысле, модели. Запомнить их номера.
   Провести ранжирование моделей по шагам. Число шагов равно $(n-1)$. Положить $t=0$.
   Положить $t=t+1$.
3. Найти числа $H^{(t)}_l$, $M^{(t)}_l$, $C^{(t)}_{klmax}$ и определить лучшую модель S_j с минимальными значениями $H^{(t)}_l$. Если таких моделей несколько, то лучшей среди них будет модель с максимальным значением $M^{(t)}_l$. Если число моделей с равными максимальными значениями $M^{(t)}_l$ более единицы, то лучшей будет модель с минимальным значением $C^{(t)}_{klmax}$.
   Если лучшие модели имеют соответственно равные значения $H^{(t)}_l$, $M^{(t)}_l$, $C^{(t)}_{klmax}$, то такие модели считают эквивалентными.
   Запомнить номер j, занести его в кортеж Парето P.
   Исключить из оценочной матрицы j-ю строку j-й столбец.
   Если $t<n-1$, перейти к шагу 5, иначе – к шагу 10.
   Занести в множество Парето P оставшуюся модель.
4. Если задача решается для одной группы коэффициентов важности – перейти к шагу 13, иначе – к шагу 12.
5. Если задача решается для множества групп коэффициентов важности, повторить шаги 2-10 нужное число раз. Перейти к шагу 13.
6. Провести анализ результатов решения. При необходимости повторить шаги 2-12. Перейти к шагу 14.
7. Конец решения.

Методика решения задачи гипервекторного ранжирования

1. Провести анализ исходной информации, формирование критериев оценок моделей, определить коэффициенты важности критериев.
2. Провести ранжирование моделей по множеству скалярных компонент каждой векторной компоненты.
3. Определить псевдозначения векторных компонент.
4. Провести ранжирование моделей по множеству векторных компонент (построить частные кортежи Парето).
5. Определить псевдозначения многовекторных компонент.
6. Построить кортеж Парето.
7. Провести анализ результатов решения.
8. В случае необходимости уточнить исходные данные и (или) изменить коэффициенты важности критериев. Перейти к шагу 2. В противоположном случае перейти к шагу 9.
9. Конец решения.

6. Выводы

Таким образом, рассмотрена важная в прикладном плане задача гипервекторного ранжирования моделей разработки ПО.

Основу метода гипервекторного ранжирования составляет метод «жесткого ранжирования», который последовательно применяется $\left[E + 1 + \sum\limits_{\varepsilon=1}^{E} r_\varepsilon \right]$ раз.

Решение задачи оказалось возможным благодаря введению псевдозначений векторных и многовекторных компонент. Более подробно методы ранжирования рассмотрены в [17].

DOI: 10.21681/2311-3456-2021-5-21-29

Литература

1. Федорец О.Н., Грибков В.В. Модели и методы разработки безопасного программного обеспечения. Монография. – М.: Типография «Славянская Поляна», 2009. 187 с.
2. Буйневич М.В., Израилов К.Е. Аналитическое моделирование работы программного кода с уязвимостями // Вопросы кибербезопасности. 2020. № 3 (37). С. 2-12. DOI: 10.21681/2311-3456-2021-3-02-12
3. Казарин О. В., Скиба В. Ю. Реализация практически значимых проактивно безопасных компьютерных систем // Защита информации. Инсайд. 2015. № 5. С. 40-43.
4. Методы оценки несоответствия средств защиты информации / А.С.Марков, В.Л.Цирлов, А.В.Барабанов. М.: Радио и связь, 2012. 192 с.
5. Петренко А.С., Петренко С.А. Безопасная Agile-разработка системы ЭДО // Защита информации. Инсайд. 2018. № 3 (81). С. 30-35.
6. Соловьёв В.С., Язов Ю.К. Информационное обеспечение деятельности по технической защите информации // Вопросы кибербезопасности. 2021. №1(41). С.69-79. DOI: 10.21681/2311-3456-2021-1-69-79
7. Забродин А.А., Макарец А.Б. Преимущества и недостатки использования гибких методологий разработки программного обеспечения Microsoft solutions framework (MSF), Rational unified process (RUP) и Extreme programming / В сборнике: Математика и математическое моделирование. Сборник материалов XIII Всероссийской молодежной научно-инновационной школы. 2019. С. 335-336.
8. Никулина А.С., Тынянова С.Е., Голубева А.И. Понятие и основные принципы RUP и UML // Аллея науки. 2018. Т. 2. № 7 (23). С. 707-712.
9. Brown S. Software Architecture for Developers. Category: Computer science Software development technology. Leanpub, 2015. 275 p.
10. Синицын С.В., Налютин Н.Ю. Верификация программного обеспечения: учебное пособие. М.: Интернет-Университет Информационных технологий; БИНОМ. Лаборатория знаний, 2008.
11. Фатрелл Р.Т., Шафер Д.Ф., Шафер Л.И. Управление программными проектами: достижение оптимального качества при минимуме затрат. Пер. с англ. М.: Издательский дом «Вильямс», 2003.
12. Бахтизин В.В., Глухова Л.А. Выбор модели жизненного цикла разработки программных средств и систем на основе сводной таблицы критериев классификации проекта // Доклады белорусского государственного университета информатики и радиоэлектроники. 2005. январь – март, №1. С. 110-113.
13. Барабанов А.В., Марков А.С., Цирлов В.Л. 28 магических мер разработки безопасного программного обеспечения // Вопросы кибербезопасности. 2015. № 5 (13). С. 2-10.
14. Barabanov A., Markov A. Modern trends in the regulatory framework of the information security compliance assessment in Russia based on Common Criteria / В сборнике: ACM International Conference Proceeding Series. Proceedings of the 8th International Conference on Security of Information and Networks, SIN 2015. 2015. С. 2799980
15. Probabilistic Modeling in System Engineering / By ed. A. Kostogryzov — London: IntechOpen, 2018. 278 p. 10.5772/ intechopen.71396. DOI: 10.5772/intechopen.71396
16. Ряполова Е.И., Шрейдер М.Ю., Боровский А.С. Метод обработки информации для поддержки принятия решений в управлении облачным сервисом // Вопросы кибербезопасности. 2018. № 3 (27). С. 39-46. DOI: 10.21681/2311-3456-2018-3-39-46
17. Сафронов В.В. Основы системного анализа: методы многовекторной оптимизации и многовекторного ранжирования: Монография. – Саратов: Научная книга, 2009. 329 с.

CHOOSING A RATIONAL MODEL FOR DEVELOPMENT SECURE SOFTWARE

Zhidkov I.V.[4], Zubarev I.V. [5], Khabibullin I.V.[6]

Objective: *the purpose of the study is to justify approaches to selecting a rational software development model based on the proposed criteria system.*

Methods: *during the research, the methods of decision theory, the method of ranking systems by the set of scalar components, in particular, the method of "hard" ranking, were used.*

Study result: *analysis of the main models of software development was carried out, a system of criteria used to compare different models of software development was considered, proposals for selecting a rational model of software development based on solving the problem of hypervector ranking were developed.*

Keywords: *software, software lifecycle, development model, ranking method.*

4 Igor Zhidkov, Ph.D., Associate Professor, FSBI «3 Central Research Institute» of the Ministry of Defense of Russia, Moscow, Russia. E-mail: igorzh@bk.ru

5 Igor Zubarev, Ph.D., Associate Professor, Head of Department, Techmash, Moscow, Russia. E-mail: i.v.zubarev@tecmash.ru

6 Ilfat Khabibullin, Russian Defense Ministry employee, Moscow, Russia. E-mail: rainaira1632@mail.ru

References

1. Fedoretc O.N., Gribkov V.V. Modeli i metody` razrabotki bezopasnogo programmnogo obespecheniia. Monografiia. – M.: Tipografiia «Slavianskaia Poliana», 2009. 187 s.
2. Bui`nevich M.V., Izrailov K.E. Analiticheskoe modelirovanie raboty` programmnogo koda s uiazvimostiami // Voprosy` kiberbezopasnosti. 2020. № 3 (37). S. 2-12. DOI: 10.21681/2311-3456-2021-3-02-12
3. Kazarin O. V., Skiba V. Iu. Realizatciia prakticheski znachimy`kh proaktivno bezopasny`kh komp`iuterny`kh sistem // Zashchita informatcii. Insai`d. 2015. № 5. S. 40-43.
4. Metody` ocenki nesootvetstviia sredstv zashchity` informatcii / A.S.Markov, V.L.Tcirlov, A.V.Barabanov. M.: Radio i sviaz`, 2012. 192 s.
5. Petrenko A.S., Petrenko S.A. Bezopasnaia Agile-razrabotka sistemy` E`DO // Zashchita informatcii. Insai`d. 2018. № 3 (81). S. 30-35.
6. Solov`yov V.S., Iazov Iu.K. Informatcionnoe obespechenie deiatel`nosti po tekhnicheskoi` zashchite informatcii // Voprosy` kiberbezopasnosti. 2021. №1(41). S.69-79. DOI: 10.21681/2311-3456-2021-1-69-79
7. Zabrodin A.A., Makaretc A.B. Preimushchestva i nedostatki ispol`zovaniia gibkikh metodologii` razrabotki programmnogo obespecheniia Microsoft solutions framework (MSF), Rational unified process (RUP) i Extreme programming / V sbornike: Matematika i matematicheskoe modelirovanie. Sbornik materialov XIII Vserossii`skoi` molodezhnoi` nauchno-innovatcionnoi` shkoly`. 2019. S. 335-336.
8. Nikulina A.S., Ty`nianova S.E., Golubeva A.I. Poniatie i osnovny`e printcipy` RUP i UML // Alleia nauki. 2018. T. 2. № 7 (23). S. 707-712.
9. Brown S. Software Architecture for Developers. Category: Computer science Software development technology. Leanpub, 2015. 275 p.
10. Sinitcy`n S.V., Naliutin N.Iu. Verifikatciia programmnogo obespecheniia: uchebnoe posobie. M.: Internet-Universitet Informatcionny`kh tekhnologii`; BINOM. Laboratoriia znanii`, 2008.
11. Fatrell R.T., Shafer D.F., Shafer L.I. Upravlenie programmny`mi proektami: dostizhenie optimal`nogo kachestva pri minimume zatrat. Per. s angl. M.: Izdatel`skii` dom «Vil`iams», 2003.
12. Bakhtizin V.V., Gluhova L.A. Vy`bor modeli zhiznennogo tcicla razrabotki programmny`kh sredstv i sistem na osnove svodnoi` tablitcy` kriteriev classifikatcii proekta // Doclady` belorusskogo gosudarstvennogo universiteta informatiki i radioe`lektroniki. 2005. ianvar` – mart, №1. C. 110-113.
13. Barabanov A.V., Markov A.S., Tcirlov V.L. 28 magicheskikh mer razrabotki bezopasnogo programmnogo obespecheniia // Voprosy` kiberbezopasnosti. 2015. № 5 (13). S. 2-10.
14. Barabanov A., Markov A. Modern trends in the regulatory framework of the information security compliance assessment in Russia based on Common Criteria / V sbornike: ACM International Conference Proceeding Series. Proceedings of the 8th International Conference on Security of Information and Networks, SIN 2015. 2015. S. 2799980
15. Probabilistic Modeling in System Engineering / By ed. A. Kostogryzov — London: IntechOpen, 2018. 278 p. 10.5772/ intechopen.71396. DOI: 10.5772/intechopen.71396
16. Riapolova E.I., Shrei`der M.Iu., Borovskii` A.S. Metod obrabotki informatcii dlia podderzhki priniatiia reshenii` v upravlenii oblachny`m servisom // Voprosy` kiberbezopasnosti. 2018. № 3 (27). S. 39-46. DOI: 10.21681/2311-3456-2018-3-39-46
17. Safronov V.V. Osnovy` sistemnogo analiza: metody` mnogovektornoi` optimizatcii i mnogovektornogo ranzhirovaniia: Monografiia. – Sarahtov: Nauchnaia kniga, 2009. 329 s.