УДК 004.41:004.056

А.А. Смирнов, А.В. Коваленко, Н.Н, Якименко, А.П. Доренский

Кировоградский национальный технический университет, Кировоград

МЕТОД КАЧЕСТВЕННОГО АНАЛИЗА РИСКОВ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Данная статья посвящена формированию метода качественного анализа рисков разработки программного обеспечения. Его отличительной особенностью является учет факторов эксплуатационных рисков, особенно риска невыявления уязвимостей программного обеспечения и оценка произвольного непротиворечивого конечного набора «квантов информации».

Ключевые слова: *оценка рисков, разработка программного обеспечения*

Постановка проблемы исследования

Авторами разработан комплекс методов качественного анализа и количественной оценки рисков разработки программного обеспечения, что позволило решить противоречие, возникающих при разработке программного обеспечения (ПО), и заключающееся в пренебрежении фирмами-разработчиками ПО факторов уязвимости безопасности ПО. В качестве решения указанной проблемы предложено использование разработанных методов качественного анализа и количественной оценки рисков разработки программного обеспечения.

Анализ литературы [1-16] и проведенные исследования показали, что общая последовательность оценки рисков чаще всего включает в себя следующие действия:

1. Выявление источников и причин риска разработки ПО, этапов и работ, при выполнении которых возникает риск.
2. Идентификация всех возможных рисков, свойственных рассматриваемому проекту.
3. Документирование результатов и их последующая приоритезация.
4. Оценка уровня отдельных рисков и риска проекта в целом, определяющая его экономическую целесообразность.
5. Определение допустимого уровня риска разработки ПО.
6. Разработка мероприятий по снижению риска.

В соответствии с данным алгоритмом оценка риска подразделяется на три взаимно дополняющих направления: качественный (этапы 1, 2, 3) и количественный анализ (этапы 4, 5) рисков разработки ПО, а также управление (этап 6).

Данная статья посвящена разработке метода качественного анализа рисков разработки программного обеспечения.

Проведенные исследования показали, что методика качественной оценки рисков проекта является описательной, и представляет собой процесс, направленный на выявление конкретных рисков проекта, а так же порождающих их причин, с последующей оценкой возможных последствий и выработку мероприятий по работе с рисками. В процессе качественного анализа рисков происходит выработка метрик, отвечающих за определение граничных показателей факторов, символизирующих о проявление риска/ов.

Выявление источников и причин риска разработки ПО, этапов и работ, при выполнении которых возникает риск

Рассматривая первый пункт, приведенного выше перечня действий по качественному и количественному анализу рисков, заметим, что исходные данные для выявления и описания характеристик рисков могут браться из разных источников:

- база знаний организации;
- информация из открытых источников, научных работ;
- маркетинговая аналитика;
- опрос экспертов и др.

Ряд известных авторов [1-16], проведя исследования, выявили наиболее распространенный риски при разработке ПО. Например, авторы Демарко и Листер [1-5, 8] приводят свой список из пяти наиболее важных источников рисков любого проекта разработки ПО:

- изъяны календарного планирования;
- текучесть кадров;
- раздувание требований;
- нарушение спецификаций;
- низкая производительность.

Можно отметить, что данный перечень имеет обобщенный характер, что в значительной степени затрудняет метрическую оценку приведенного списка.

Барии Боэм в своей работе [6] расширяет список до 10 наиболее распространенных рисков программного проекта:

1. Дефицит специалистов.

2. Нереалистичные сроки и бюджет.

3. Реализация несоответствующей функциональности.

4. Разработка неправильного пользовательского интерфейса.

5. "Золотая сервировка", перфекционизм, ненужная оптимизация и оттачивание деталей.

6. Непрекращающийся поток изменений.

7. Нехватка информации о внешних компонентах, определяющих окружение системы или вовлеченных в интеграцию.

8. Недостатки в работах, выполняемых внешними (по отношению к проекту) ресурсами.

9. Недостаточная производительность получаемой системы.

10. "Разрыв" в квалификации специалистов разных областей знаний.

Однако и этот перечень не полный, и неструктурированный. Это затрудняет процесс оценки взаимовлияния приведенных рисков друг на друга.

Достаточно подробно риски были оценены и классифицированы в работах [1-16]. В соответствии с данными исследованиями риски классифицируются по следующим признакам:

- среда (внутренний, внешний риски);
- природа (экономический, технический, технологический);
- сфера (риск проекта, процесса, продукта);
- уровень (от критического к незначительному риску);
- отрасль воздействия (риск невыполнения бюджета проекта, риск невыполнения плана проекта, риск невыполнения качества проекта);
- звено управления риском (риск отдельного процесса, риск проекта, риск компании).

Однако подобная классификация делает акцент на проектах разработки программных систем, которые не связаны с процессами их дальнейшего внедрения и адаптации систем в условиях конкретной организации, эксплуатации в условиях возможных внешних злоумышленных воздействий. Поэтому представляется целесообразным необходимость рассматривать отдельно:

- организационные риски, которые связаны с тем, что проект вызовет такие изменения в структуре и бизнес-процессах кампании, которые нивелируют запланированные выгоды;
- операционные риски, связанные с неконтролируемым ростом затрат на эксплуатацию системы;
- социальные риски, связанные с неадекватным поведением участников проекта;
- эксплуатационные риски, связанные с возможными будущими финансовыми, имиджевыми и другими потерями в случае наличия потенциальных уязвимостей проектов.

Методика структурной идентификации рисков разработки программного обеспечения

Используя результаты исследований приведенных выше авторов [1-16], мнения экспертов, маркетинговые данные, а также базы знаний таких известных фирм как *Epam Systems* и *Nix Solutions Ltd*, идентифицируем риски разработки ПО и представим результат в виде структурной схемы классификации рис. 1.

Как видно из рис.1. основные риски разработки программного обеспечения можно представить в виде совокупности множеств:

организационных $Z = \{\mathrm{Id}\,1, \ldots, \mathrm{Id}\,5\}$,

управленческих $U = \{\mathrm{Id}\,6, \ldots, \mathrm{Id}\,9\}$,

операционных $Y = \{\mathrm{Id}\,10, \ldots, \mathrm{Id}\,15\}$,

технологических $T = \{\mathrm{Id}\,16, \ldots, \mathrm{Id}\,20\}$,

эксплуатационных $E = \{\mathrm{Id}\,21, \ldots, \mathrm{Id}\,24\}$,

социальных $C = \{\mathrm{Id}\,25, \ldots, \mathrm{Id}\,27\}$,

правовых $W = \{\mathrm{Id}\,28, \mathrm{Id}\,29\}$ рисков.

Отличительной особенностью представленной классификации является учет эксплуатационных рисков. Особенную важность эти риски приобретают в условиях повышенного уровня киберпреступности, когда пренебрежение уязвимостями программного обеспечения может привести к эксплуатационным проблемам, а зачастую и невозможности эксплуатации («краху») ПО.

Кроме этого, в условиях украинского правового поля наблюдаются отдельные случаи неадекватности и несоответствия правовым нормам действий должностных лиц государственного аппарата.

Практика ряда известных фирм-разработчиков ПО (*Nix Solutions Ltd*, и др.) показывает, что указанный фактор риска целесообразно учитывать при разработке ПО, наряду с фактором возможного изменения украинского законодательства.

Влияние указанных на рис. 1 рисков на основные факторы успеха разработки, внедрения и длительной эксплуатации ПО проиллюстрировано на рис. 2.

Как видно из этого рисунка большинство из рассматриваемых рисков разработки ПО (организационные, операционные, управленческие и др.) могут оказывать непосредственное влияние как на процесс разработки ПО, так и на процесс его эксплуатации. В то же время, например, эксплуатационные риски непосредственного влияния на процесс разработки ПО не оказывают. Но пренебрежение этими рисками ведет зачастую к провалу эксплуатации ПО и потерям будущих заказов и проектов (простоям разработчиков ПО). Именно этим фактором вызвана связь между блоками «Провал при эксплуатации ПО» и «Провал при разработке ПО».

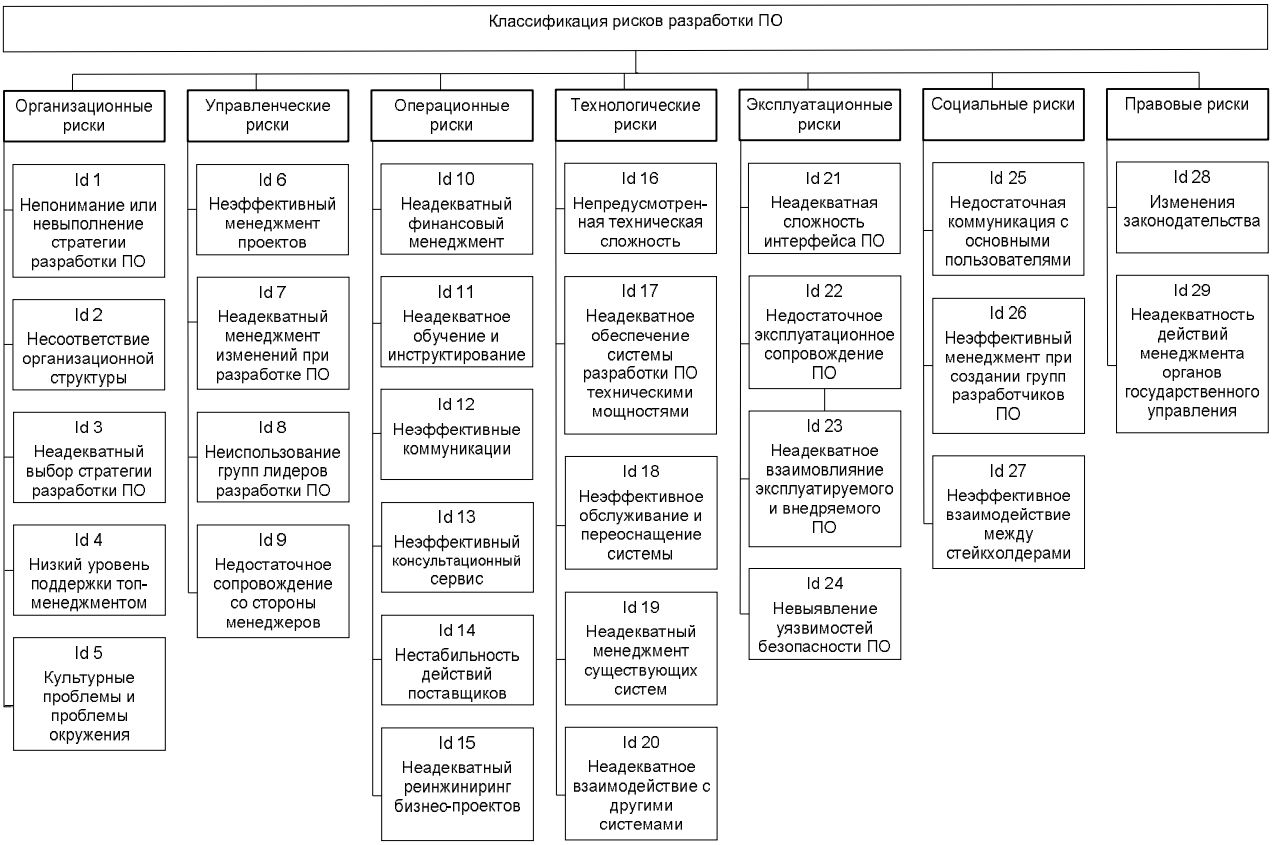

Рис. 1. Классификация рисков разработки программного обеспечения

Однако, несмотря на это в целом можно выделить множество рисков непосредственно влияющих на процесс разработки ПО:

$$MR = \{Z, U, Y, C, T, W\},$$

и множество рисков непосредственно влияющих на процесс эксплуатации ПО:

$$ME = \{Z, U, Y, C, T, W, E\},$$

$$(Id\,9, Id\,10,\ Id\,15, Id\,29 \notin ME).$$

Следует заметить, что выделенные на рис. 2. факторы в достаточной степени описывают перечень возможных рисков разработки ПО. Однако, они не дают представления о взаимном влиянии и соответственно возможном изменении конечного результата. Кроме этого приведенные множества рисков разработки ПО в разной степени влияют на конечный результат.

Поэтому следующим шагом идентификации рисков разработки ПО целесообразно выполнить процедуры ранжирования и выделения наиболее приоритетных (важных) рисков разработки программного обеспечения.

Проведенные исследования показали, что для решения задачи определения взаимовлияния рисков целесообразно использовать инструмент анализа причинно-следственных связей между различными факторами и рисками, разработанный Каору Исикава [7] (диаграмма Исикавы). В соответствии с известным принципом Парето [7], среди множества потенциальных причин (причинных факторов, по Исикаве), порождающих проблемы (следствие), лишь две-три являются наиболее значимыми, их поиск и должен быть организован.

Для этого осуществляется:

Изображение диаграммы Исикавы дает возможность получить более подробную информацию о возможности взаимовлияния различных видов риска друг на друга, что так же даст уточняющие данные для количественного анализа рисков. Однако задачу выбора наиболее приоритетных рисков диаграмма решить не может.

Для решения этой задачи в работе предлагается использовать математический аппарат многокритериальной оптимизации, основанной на локальной геометрии множества Парето.

Анализ литературы [3 – 6] показал, что существуют, по крайней мере, три формулировки многокритериальной оптимизации, основанной на локальной геометрии множества Парето:

1. Локальная. Найти одно Парето-оптимальное решение (ближайшее к заданной начальной точке).

2. Глобальная. Найти конечное множество Парето-оптимальных решений, достаточно хорошо описывающее (покрывающее) истинный Парето-фронт.

3. Интерактивная. Найти Парето-оптимальное решение, максимально удовлетворяющее предпочтениям лица принимающего решение (ЛПР).

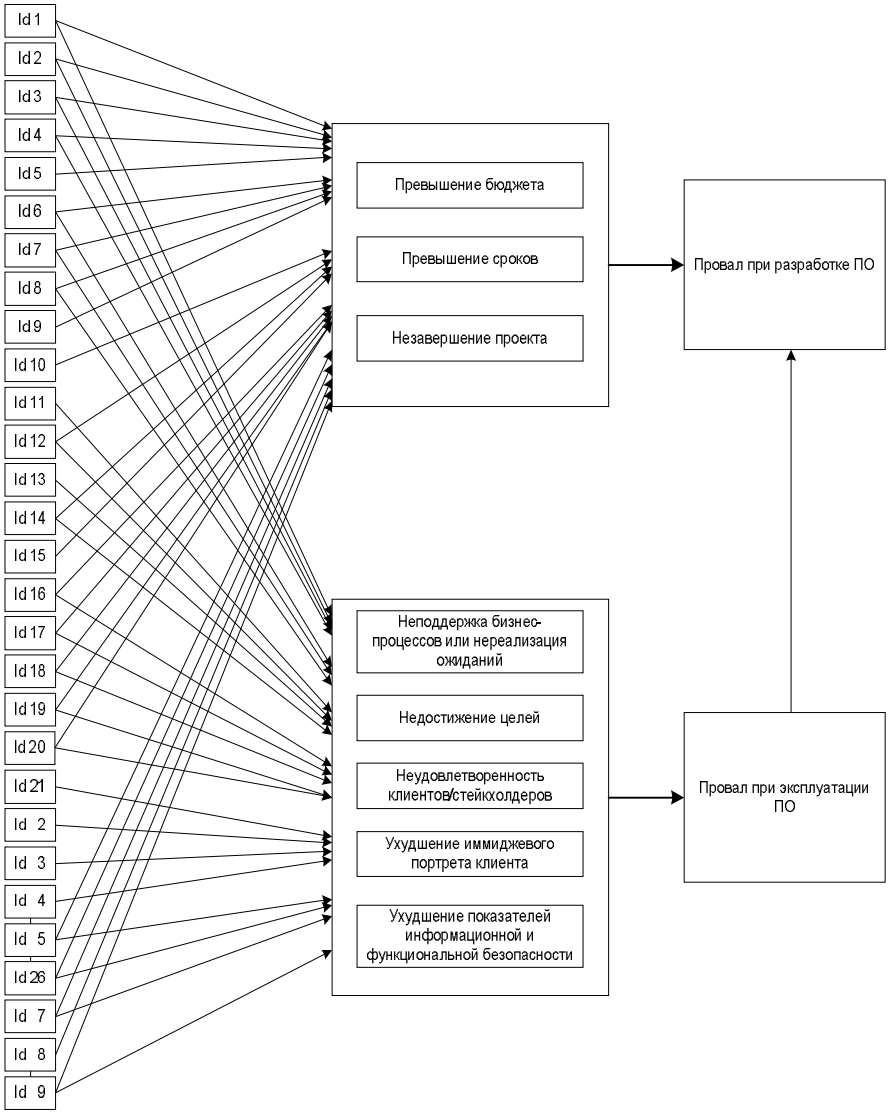

Рис. 2. Схема влияния рисков на основные факторы
успеха разработки, внедрения и длительной эксплуатации программного обеспечения

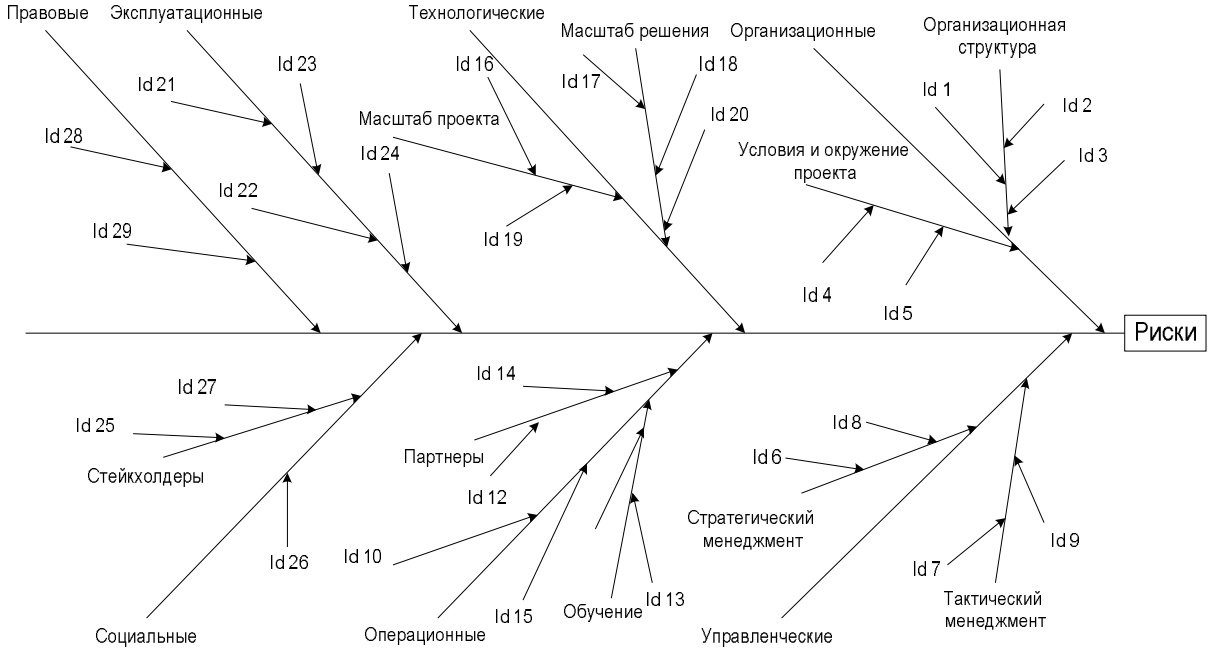

Рис. 3. Дерево решений диаграммы рисков разработки программного обеспечения

Проведенные исследования показали, что в процессах, построенных на принципах постоянных коммуникаций между участниками, использования «мозговых штурмов» с привлечением мнений экспертов, целесообразным представляется использование интерактивной формулировки многокритериальной оптимизации.

В этих условиях абстрактную задачу выбора наиболее важных рисков разработки ПО из имеющегося исходного множества возможных (допустимых) вариантов (решений) X можно сформулировать следующим образом.

Обозначим множество всех заранее определенных рисков разработки ПО через S(X). Очевидно, S(X)<X. Таким образом, в задаче выбора дано множество X, содержащее, по крайней мере, два элемента, а требуется найти некоторое его непустое подмножество S(X). Предполагается, что выбор производится ЛПР, в роли которого может выступать как отдельный человек, так и целый коллектив разработчиков. Для того, чтобы совершаемый выбор в наибольшей степени соответствовал достижению имеющейся цели (т.е. был «наилучшим» или «оптимальным» для данного ЛПР), необходимо в процессе выбора учитывать мнение экспертов.

Проведенные исследования показали, что в настоящее время существует множество подходов учета мнения экспертов (метод анализа иерархий [], реализованный в программном продукте EXPERT CHOICE, метод «искусственного» отношения предпочтения [], и др.) однако все они обладают существенными недостатками, главный из которых заключается в том, что, несмотря на многообразие и детальную изученность иерархий и «искусственных» отношений, крайне редко какое-либо из них можно считать удовлетворяющим конкретное ЛПР в полной мере. Характерным примером, подтверждающим данный факт является пренебрежение оценкой уязвимостей разработанного ПО (недостаточность или полное отсутствие pen-тестирования).

Поэтому для решения задачи выбора наиболее приоритетных рисков (сужение множества Парето) предлагается использовать «кванты информации».

Для этого рассмотрим произвольные оценки рисков разработки ПО

$$y' = (y_1', \ldots, y_m')$$

и

$$y'' = (y_1'', \ldots, y_m''),$$

принадлежащие множеству парето-оптимальных векторов $f(P_f(X))$.

По определению множества Парето должны найтись такие два непустых подмножества номеров критериев

$$A, B \subset I = \{1,2,\ldots,m\},$$

что

$$y_i' > y_i'', \quad y_i' - y_i'' = w_i > 0, \quad \forall i \in A, \qquad (1)$$

$$y_j'' > y_j', \quad y_j'' - y_j' = w_j > 0, \quad \forall j \in B, \qquad (2)$$

$$y_s'' = y_s', \quad \forall s \in I \setminus (A \cup B). \qquad (3)$$

Согласно условиям (1-3), первый вектор превосходит второй по компонентам группы критериев A, тогда как второй превосходит первый по компонентам группы критериев B. По остальным компонентам (если таковые имеются) два указанных вектора совпадают.

Сужение множества Парето, т.е. удаление некоторых парето-оптимальных векторов, обычно происходит на основе сравнения. Человеку проще всего сравнивать пары.

Если при сравнении фиксированной пары парето-оптимальных векторов y′ и y″ вида (1-3) ЛПР «выбраковывает» один из этих векторов (например, второй), то это означает, что для него первый вектор предпочтительнее второго, т.е. $y' \succ y''$, где $\succ$ – отношение предпочтения, определённое на всём критериальном пространстве $\Re^m$ и совпадающее на множестве Y с отношением $\succ$ y .

Соотношение $y' \succ y''$, задаёт «квант информации» об отношении строгого предпочтения, который свидетельствует о готовности ЛПР к компромиссу – оно согласно пойти на потери по всем критериям группы B в размере w_j ради того, чтобы получить прибавки в размере w_i по критериям группы A, сохранив при этом значения всех остальных критериев.

Наличие указанного «кванта информации» позволяет сократить множество Парето на один вектор y″. Для того чтобы добиться большего сокращения, можно принять, что $y' \succ y''$, имеет место не только для данной пары векторов, но и для всех тех векторов, которые удовлетворяют условиям (1-3) при неизменных значениях w_i и w_j.

В этом случае предложено говорить, что группа критериев A важнее группы B. При указанном расширении действия «кванта информации» можно рассчитывать на более заметное сужение множества Парето, хотя нередко и оно оказывается недостаточным для окончательного выбора. В таких случаях имеет смысл наложить дополнительные требования на отношение предпочтения так, чтобы действие «кванта информации» в сужении множества Парето оказалось более эффективным.

Эти требования (без аксиомы исключения) сформулированы в [1 – 11]. Позднее было установлено, что они представляют собой дальнейшее усиление системы двух упоминавшихся ранее аксиом, гарантирующих выполнение принципа Эджворта-Парето.

Аксиома 1 (аксиома исключения).

Аксиома 2. Отношение $\succ$ определено на всём критериальном пространстве $\Re^m$ и является транзитивным на нём.

Аксиома 3 (аксиома согласования). Из двух векторов, отличающихся один от другого единственной компонентой, для ЛПР предпочтительнее вектор, имеющий большую компоненту.

Аксиома 4 (аксиома инвариантности). Отношение предпочтения $\succ$ инвариантно относительно линейного положительного преобразования (т.е. является линейным).

Пусть один критерий (или группа критериев) важнее другого критерия (другой группы критериев), если имеет место некоторое условие Ξ, которое содержит определённую информацию об отношении предпочтения ЛПР.

Отсюда ясно, что без определения важности критериев всегда можно обойтись, оперируя в процессе принятия решений непосредственно с условием Ξ.

Чтобы воспользоваться определением важности, основанном на «кванте информации» и использующем в качестве условия Ξ соотношения (1-3), сначала следует объяснить ЛПР это определение важности, убедиться, что оно его «усвоило», после чего для выявления предпочтений ЛПР задать ему вопрос на «языке важности»: является ли группа критериев A важнее группы B с параметрами w_i и w_j (для $i \in A$ и $j \in B$).

Из [8] известно, что бинарное отношение $\succ$, заданное на векторном пространстве $\Re^m$, называется конусным, если существует такой конус $K \subset \Re^m$, что соотношение $y' \succ y''$ имеет место тогда и только тогда, когда $y' - y'' \in K$.

Аксиома 5. Любое бинарное отношение $\succ$, заданное на векторном пространстве $\Re^m$ и удовлетворяющее аксиомам 2−4, является конусным с острым выпуклым конусом (без начала координат), который содержит все векторы с неотрицательными компонентами. Обратно, всякое конусное отношение $\succ$ с указанным конусом удовлетворяет аксиомам 2−4.

Аксиома 5 открывает возможность использования аппарата выпуклого анализа и построения содержательной математической теории для учёта различного набора «квантов информации». Наиболее простой случай одного «кванта» рассматривается в следующем утверждении, доказательство которого опирается на факты из теории двойственности выпуклого анализа.

Аксиома 6. Пусть выполнены аксиомы 2–4 и имеется «квант информации» об отношении предпочтения $\succ$. Тогда для любого множества выбираемых вариантов S(X), удовлетворяющего аксиоме 1, справедливы включения

$$S(X) \subset P_g(X) \subset P_f(X),$$

причём «новый» векторный критерий g может быть образован из функций f_i для всех $i \in I \setminus B$:

$$g_{i,j} = w_j f_i + w_i f_j \text{ для всех } i \in A, j \in B, \qquad (4)$$

либо из функций f_i для всех $i \in I \setminus B$:

$$f_0 = \min_{i \in A} \frac{f_i}{w_i} + \min_{j \in B} \frac{f_j}{w_j}. \qquad (5)$$

Важная особенность аксиомы 6 заключается в отсутствии каких-либо требований к множеству X и векторному критерию f: эти объекты могут быть любыми.

Ограничения накладываются лишь на поведение ЛПР в процессе принятия решений и выражаются они в форме аксиом 1–4.

Аксиомой 6 указывается оценка сверху $P_g(X)$ для неизвестного множества выбираемых вариантов S(X), более точная, чем множество Парето $P_f(X)$. Сама оценка представляет собой множество парето-оптимальных вариантов, но относительно «нового» векторного критерия g.

Для того, чтобы сформировать g, из «старого» векторного критерия f следует удалить все компоненты группы критериев B и добавить один нелинейный критерий f_0 вида (5), либо $|A| \cdot |B|$ «новых» линейных критериев вида (4), где $|L|$ обозначает число элементов конечного множества L.

Вариант с нелинейной функцией f_0 вида (5) можно применять для количественных критериев, значения которых измеряются в шкале отношений, тогда как вариант (4) допускает использование ещё и в шкале интервалов.

Нелинейную функцию f_0 вида (5) можно использовать для изучения случая, когда одна группа критериев важнее другой, где в отличие от приведённой выше аксиоматики используется операция транзитивного замыкания бинарного отношения и некоторые другие предположения.

Как показали исследования учёт нескольких «квантов информации» должен в большей степени способствовать сужению множества Парето. Однако, некоторые может случиться ситуация, когда ряд «квантов информации» будут иметь противоречивый смысл, и их использование будет невозможным. Поэтому важной является задача выбора непротиворечивых «квантов информации».

В рамках работы под непротиворечивым названо такое множество, которое «порождает» иррефлексивное отношение.

Построение оценки сверху для неизвестного множества выбираемых векторов S(Y)=f(S(X)) в виде множества

$$\overline{P}(Y) = f(P_g(X))$$

при наличии произвольного непротиворечивого конечного набора «квантов информации» в случае конечного множества Y сводится к последовательной проверке соотношения

$$y' \succ_m y'' \qquad (6)$$

для всех пар допустимых векторных оценок $y', y'' \in Y$, где $\succ_m$ – бинарное отношение, которое строится на основе имеющегося непротиворечивого множества «квантов информации».

Таким образом, получила дальнейшее развитие методика структурной идентификации рисков разработки ПО, отличающаяся от известных построением оценки рисков разработки ПО «сверху» в виде множества, при наличии произвольного непротиворечивого конечного набора «квантов информации».

Используя, приведенную выше методику проведем оценку ранга рисков разработки ПО.

Исследование разработанной методики структурной идентификации рисков

После того, как риски разработки ПО выявлены и включены в реестр рисков, возникает необходимость оценки и определения их ранга отдельно для каждой цели процесса/проекта (например, для рамок функциональности, времени или других ресурсов), и построения матрицы вероятности и последствий [2]. Ранг риска позволяет оперативно управлять реагированием на риски, расположенные в различных зонах матрицы. Зоны матрицы играют роль приоритетов. Как было указано на принятие решения о ранге риска влияют приоритеты ЛПР сформированные во многом на основе экспертных оценок или результатов мозгового штурма (характерно для гибких методик разработки ПО). Учетом этих факторов построим матрицу качественной оценки ранга рисков разработки ПО в соответствии с данными рис. 2 и экспертными оценками специалистов ряда известных фирм-производителей ПО (Nix Solutions LTD, Epam Systems) [1-5].

В табл. 1 приведены результаты качественной оценки ранга рисков разработки ПО. Следует заметить, что зоны матрицы играют роль приоритетов. К примеру, для рисков, расположенных в зоне высокого риска (выделено темно-серым цветом и составляют множество D) матрицы необходимы предупредительные операции и агрессивная стратегия реагирования. Для угроз, расположенных в зоне низкого риска (выделено наклонным шрифтом и составляют множество G), осуществление предупредительных операций может не потребоваться, если держать под контролем все содержание выполняемой деятельности. В свою очередь множество угроз среднего риска (полужирный шрифт, множество F) требуют обязательной стратегии управления и реагирования. Как видно из табл. 1. основная часть организационных, операционных, управленческих и эксплуатационных рисков находится в «закрашенной» зоне. Это говорит о важности учета данных рисков (особенно в современных условиях применения гибких методологий разработки ПО).

Следует заметить, что многие риски (например, Id 18, Id 20), в начале определенной активности могут находиться в зоне низкого ранга, а ближе к ответственным вехам переместиться в пограничные или более критичные зоны. В то же время ряд существующих рисков вне зависимости от первоначального уровня ранга могут переместиться в более «критичную» область (например, Id 23, Id 24 и др.).

Таким образом, предложенный аппарат идентификации и качественной оценки ранга рисков разработки ПО позволяет до 17% сузить множество важных рисков и, соответственно, первоочередных решений управления.

Таблица 1

Результаты качественной оценки ранга рисков разработки ПО

Качественная оценка вероятности причинения вреда	Тяжесть последствий при причинении вреда				
	Очень высокая тяжесть	Высокая тяжесть	Средняя тяжесть	Низкая тяжесть	Незначительная тяжесть
Высокая вероятность	Id 1	Id 6, Id 7, Id 24,	Id 15	**Id 23, Id 27**	**Id 25**
Средняя вероятность	Id 19	Id 3, Id 10, Id 16,	**Id 4, Id 8**	**Id 21**	*Id 22*
Низкая вероятность	Id 9	**Id 2, Id 17**	**Id 12, Id 14**	*Id 18*	*Id 28*
Малая вероятность	**Id 26**	**Id 11**	*Id 20, Id 29*	*Id 13*	*Id 5*

Документирование результатов и их последующая приоритезация

Следующим этапом качественного анализа риска является процесс документирования.

Процесс анализа рисков следует документировать на протяжении жизненного цикла всего проекта/процесса. Объем документирования и его форма, содержащая результаты анализа, зависит от конкретных целей проведенного анализа риска.

Анализ документации известных фирм разработчиков ПО показал, что в итоговом документе целесообразно фиксировать следующие данные:

- титульный лист;
- список участников процесса качественного анализа рисков разработки ПО;
- аннотацию;
- содержание (оглавление);
- цели и задачи проведенного качественного анализа рисков разработки ПО;
- описание анализируемого объекта;
- методологию качественного анализа рисков разработки ПО – исходные предположения и ограничения, определяющие пределы анализа риска;
- описание используемых методов анализа и обоснование их применения;
- исходные данные и их источники;
- результаты идентификации;
- результаты качественного анализа риска;
- анализ неопределенностей результатов оценки риска;
- рекомендации по работе с рисками;
- заключение;
- перечень используемых источников информации.

Учитывая все описанные выше этапы анализа и оценки рисков разработки ПО общую структуру метода можно представить в виде схемы рис. 4.

Таким образом, в результате проведенных исследований на основе классификации и структурной идентификации рисков разработки ПО разработан метод качественного анализа рисков разработки программного обеспечения.

Отличительной особенностью разработанного метода является учет факторов эксплуатационных рисков, особенно риска невыявления уязвимостей ПО и оценка произвольного непротиворечивого конечного набора «квантов информации».

Это позволит до 17% сузить множество важных рисков и снизить возможные финансовые и имиджевые потери организаций-разработчиков программного обеспечения.

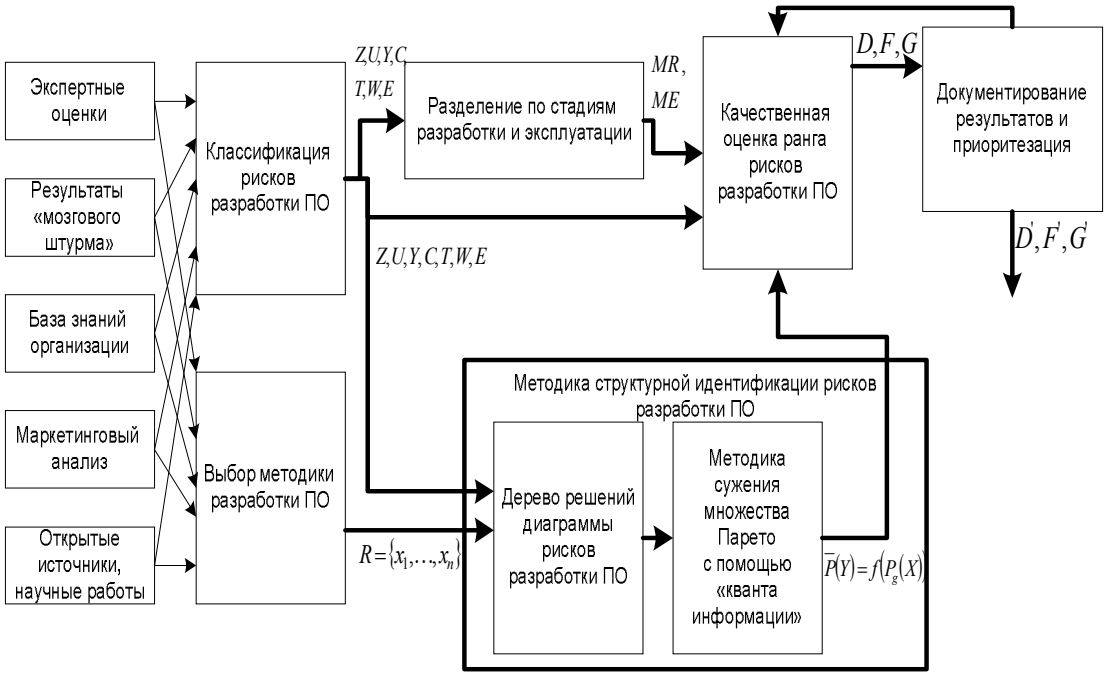

Рис. 4. Общая структура метода
качественного анализа рисков разработки программного обеспечения

Только после того, как накоплен опыт и необходимый массив данных, от качественного анализа рисков, целесообразно переходить к их количественной оценке. Причем концентрировать внимание следует именно на тех рисках, которые в процессе качественной классификации были включены в категорию высоких (особенно с высокой степенью ущерба при высокой вероятности реализации).

Выводы

В ходе решения поставленной задачи на первом этапе разработан метод качественного анализа

рисков разработки программного обеспечения. Его отличительной особенностью является учет факторов эксплуатационных рисков, особенно риска невыявления уязвимостей программного обеспечения и оценка произвольного непротиворечивого конечного набора «квантов информации».

Это позволит до 17% сузить множество важных рисков и снизить возможные финансовые и имиджевые потери организаций-разработчиков программного обеспечения.

Одной из основных составляющих метода является методика структурной идентификации рисков разработки программного обеспечения, отличающаяся от известных построением оценки рисков разработки программного обеспечения «сверху» в виде множества, при наличии произвольного непротиворечивого конечного набора «квантов информации».

Список литературы

1. Krishnan M. Soumya Software Development Risk Aspects and Success Frequency on Spiral and Agile Model / M. Soumya Krishnan // International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 3, Issue 1, January 2015 pp.301-310

2. Zeng Y. Risk Management For Enterprise Resource Planning System Implementations in Project-Based Firms : dis. for the degree of PHD / Zeng Yajun, Maryland, 2010 – pp. 210.

3. Бриткин А. И. Риски, связанные с внедрением технологий, в проектах разработки программного обеспечения / А. Бриткин // Социально-экономические и технические системы. – 2007. – № 8 (42)

4. Вишняков Я.Д Общая теория рисков: учеб. пособие для студ. высш. учеб. заведений / Я.Д.Вишняков, Н.Н.Радаев. — 2-е изд., испр. - М. : Издательский центр «Академия», 2008. – 368 с.

5. Шапкин А.С. Теория риска и моделирование рисковых ситуаций / А.С. Шапкин, В.А. Шапкиню.– М.: Издательско-торговая корпорация «Дашкв и К», 2005. – 880 с.

6. Boehm B.W. A spiral model of software development and enhancement / Boehm B., Egyed A. // IEEE Computer, May 1988 pp. 61-72

7. Исикава К. Японские методы управления качеством / К. Исикава, Сокр.пер. с англ. / Под. Ред. А. В. Гличева. — М: Экономика, 1988. — 214 с.

8. В.Д. Ногин. Принятие решений при многих критериях. Учебно-методическое пособие.– СПб. Издательство «ЮТАС», 2007. – 104 с.

9. Geymayr J. Fault-Tree Analysis: A Knowledge-Engineering Approach / J. Geymayr, N. Ebecken // IEEE Transactions on Reliability. – 1995. – № 44(1), pp. 37 – 45.

10. Анализ дерева отказов (Fault tree analysis (FTA)) / Электронный вариант Режим доступа: http://www.statistica.ru/knowledge-clusters/technical-sciences/analiz-dereva-otkazov/

11. Інженерія програмного забезпечення: Навч. посібник / [Смірнов О.А., Коваленко О.В., Мелешко Є.В. та ін.] – К.: РВЛ КНТУ, 2013. – 409 с.

12. Доренський О.П. Формалізація процесу зміни станів програмних об'єктів складних систем на основі формального апарату скінченних автоматів Мура / О.П. Доренський, О.А. Смірнов // Зв'язок. — 2014. — № 3 (109) — С. 27-31.

13. Dorensky O. Development of the theoretical bases of logical domain modeling of a complex software system / Oleksandr Dorensky, Alexey Smirnov // International Journal of Computational Engineering Research (IJCER). — India: Delhi, 2014. — Vol. 4, Issue 4. — P. 19-23

14. Лысенко И.А. Исследование уровней тестирования программного обеспечения инфотелекоммуникационных систем / И.А. Лысенко, А.А. Смирнов, Е.В. Мелешко // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 4(17). – Харків: ХУПС. – 2014. – С.79-81.

15. Лысенко И.А. Исследование процесса разработки программного обеспечения инфотелекоммуникационных систем / И.А. Лысенко, А.А. Смирнов, Л.И Полищук // Системи озброєння і військова техніка. – Випуск 4(40) – Х.: ХУПС – 2014. – С. 103-106.

16. Лысенко И.А. Исследование алгоритма выявления вида неучтенных тестовых случаев в процессе проектирования тестовых наборов / И.А. Лысенко, А.А. Смирнов // Науково-виробничій журнал "Зв'язок". - Київ: ДУТ, 2014. - № 2 (108) . – С. 153-156.

Поступила в редколлегию 15.02.2016

Рецензент: д-р техн. наук, проф. И.В. Рубан, Харьковский университет Воздушных Сил им. И. Кожедуба, Харьков.

МЕТОД ЯКІСНОГО АНАЛІЗУ РИЗИКІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

О.А. Смірнов, О.В. Коваленко, Н.М. Якименко, О.П. Доренський

Дана стаття присвячена формуванню методу якісного аналізу ризиків розробки програмного забезпечення. Його відмінною рисою є врахування чинників експлуатаційних ризиків, особливо ризику невиявлення вразливостей програмного забезпечення і оцінка довільного несуперечливого кінцевого набору «квантів інформації».

Ключові слова: *оцінка ризиків, розробка програмного забезпечення*

METHODS OF QUALITATIVE ANALYSIS RISK SOFTWARE DEVELOPMENT

A.A. Smirnov, A.V. Kovalenko, N.M. Yakimenko, O.P. Dorensky

This article deals with the formation of the method of qualitative analysis software development risks. Its distinguishing feature is the account of operational risk factors, particularly the risk of not detecting software vulnerabilities and evaluation of arbitrary finite consistent set of "quantum information".

Keywords: *risk assessment, software development.*