УДК 681.34

МНОГОАТРИБУТИВНОЕ УПРАВЛЕНИЕ ТРУДОЗАТРАТАМИ НА РАЗРАБОТКУ N-ВАРИАНТНЫХ ПРОГРАММНЫХ СИСТЕМ

Ковалев И.В., Нургалеева Ю.А., Ежеманская С.Н., Ерыгин В.Ю.

ГОУ ВПО «Сибирский государственный аэрокосмический университет им. академика М.Ф. Решетнева», Красноярск, e-mail: kovalev.fsu@mail.ru

Предложен многоатрибутивный подход к управлению трудозатратами на разработку N-вариантных программных систем, при этом каждый драйвер затрат определяет умножающий фактор, который позволяет оценить эффект действия атрибута на величину трудозатрат.

Ключевые слова: программная система, многоатрибутивный подход, управление, разработка, трудозатраты

MULTI-ATTRIBUTIVE EFFORT MANAGEMENT ON DEVELOPMENT OF N-VERSION PROGRAMMING SYSTEMS

Kovalev I.V., Nurgaleeva J.A., Ezhemanskaya S.N., Erygin V.J.

Siberian state aerospace university named by academician M.F. Reshetnev, Krasnoyarsk, e-mail: kovalev.fsu@mail.ru

The multi-attributive approach is offered for managing of effort required on the developing N-version programming systems, thus that each cost driver define multiply factor which enable to evaluate the attribute action effect on efforts.

Keywords: software system, multi-attributive approach, management, development, effort

Развитие современных технологий проектирования программного обеспечения (ПО) и высокая стоимость современных программных систем требуют эффективного управления трудозатратами на их разработку [3]. Использование с этой целью модулей COTS-сопровождения (commercial off-the-shelf) становится не только реальностью, но и необходимостью. Боэм Б.У. [1;2] предлагает использовать компоненты COTS-сопровождения ПО всякий раз, когда это возможно, в качестве соответствующей комплексной модели оценки на этапе проектирования программного обеспечения, при необходимости детализируя описание (например, используя промежуточную, детальную COTS и т.д.). В 1995 году Боэм ввел более совершенную модель COCOMO II, ориентированную на применение в программной инженерии XXI века [5]. В состав COCOMO II входят:

– модель композиции приложения;
– модель раннего этапа проектирования;
– модель этапа постархитектуры.

Для описания моделей COCOMO II требуется информация о размере программного продукта. Возможно использование LOC-оценок, объектных указателей, функциональных указателей. Как правило, более развитые модели являются многоатрибутивными [3], так как дополнительно учитывают как множество атрибутов программного проекта, так множество масштабных факторов, формирователей затрат, процедур поправок.

Благодаря применению N-вариантного подхода [6] к проектированию ПО (вводя избыточность версий программных модулей) достигается улучшение качества программ. При этом программной системе позволительно допускать ошибки, сгенерированные еще во время проектирования и разработки ПО. Однако улучшение характеристик надежности ПО с использованием избыточности требует дополнительных ресурсов. Поэтому для N-вариантных программных систем многоатрибутивное управление трудозатратами позволяет решить основной вопрос, встающий перед исследователем на этапе проектирования: каким образом, используя избыточность в структуре ПО, максимизировать надежность, не превышая ограничений по стоимостному фактору.

Факторы корректировки трудозатрат на программный проект.

Введение многоатрибутивности в постановку задачи основано на концепции, связанной с фактором корректировки трудозатрат на программный проект (Effort adjustment factor, EAF) [7]. Данная концепция заключается в том, что EAF создает эффект увеличения/уменьшения трудозатрат в зависимости от набора факторов среды (т.е. концепция поддерживает многоатрибутивность при оценке свойств проекта системы и позволяет управлять трудозатратами).

Факторы среды иногда называют факторами корректировки затрат, либо *драйверами* затрат [4]. Определение фактора-множителя EAF происходит в два этапа. На первом этапе драйверам затрат назначают числовые значения, а на втором этапе происходит пе-

ремножение драйверов затрат, в результате чего генерируется фактор корректировки трудозатрат C.

Таким образом, некоторые из атрибутов программного продукта могут изменять величину затрат на проект. Ниже перечислены атрибуты, входящие в модель COCOMO [5] и существенно влияющие на формирование модульных структур N-вариантных программных систем:

➢ требуемая надежность (RELY) – как правило, применяется в критических по отказоустойчивости системах реального времени;

➢ размер базы данных (DATA) – в основном применяется в приложениях обработки данных;

➢ сложность продукта (CPLX) – ограничения на время выполнения.

Атрибуты, связанные с аппаратными средствами

Следующие атрибуты имеют отношение к компьютерной платформе и могут применяться в качестве средства поддержки, а также при наличии работы, которая должна быть выполнена [7]:

• ограничение по времени выполнения (TIME) – применяется в том случае, когда быстродействие процессора является ограниченным;

• ограничение основного хранилища (STOR) – применяется в случае, когда размер памяти является ограниченным;

• изменяемость виртуальной машины (VIRT) – включает аппаратное обеспечение и операционную систему на целевом компьютере;

• время компьютера (TURN) – применяется при разработке.

Атрибуты проекта

Атрибуты, связанные с практикой и инструментами:

• практика современного программирования (MODR) – структурные или объектно-ориентированные технологии;

• современные инструменты программирования (TOOL) – CASE-средства отладчики и инструменты, используемые при выполнении тестирования;

• сжатие/расширение графика проекта (SCED) – минимизация степени отклонения от номинальных сроков выполнения проектных задач.

Атрибуты персонала

Некоторые атрибуты применяются для описания исполнителей работ, например:

• способность аналитика (AGAP);

• опыт в создании приложений (AEXP);

• способности программиста (PCAP);

• опыт в области виртуальных машин, включая операционную систему и аппаратное обеспечение (VEXP);

• опыт в области языков программирования, включая инструменты и практику (LEXP).

Несмотря на то, что наиболее часто с приложениями в рамках промежуточной модели COCOMO связываются указанные выше четыре категории атрибутов, при многоатрибутивном формировании модульных структур N-вариантных программных систем могут добавляться дополнительные атрибуты, в частности:

➢ изменяемость требований – некоторые из них являются ожидаемыми, однако большинство из них может представлять значительную проблему;

➢ изменяемость машины, предназначенной для разработки – нестабильные ОС, компиляторы, CASE-средства и т.д.;

➢ требования безопасности – применяются для классифицированных программ, в частности относящихся к классу N-вариантного программирования;

➢ доступ к данным – в системах, критичных по безопасности, часто бывает весьма затруднен;

➢ влияние стандартов и навязанных методов;

➢ влияние физического окружения.

Многоатрибутивное управление трудозатратами на разработку программной системы

Драйверы затрат, обеспечивающие управление трудозатратами на разработку ПО, являются многоатрибутивными и выбираются в соответствии с их общей значимостью для всех программных проектов, включая N-вариантные структуры ПО, причем они являются независимыми от размера проекта [7].

Каждый драйвер затрат определяет умножающий фактор [4], который позволяет оценить эффект действия атрибута на величину трудозатрат. Фактор EAF представляет собой произведение факторов корректировки затрат:

$$\text{EAF} = C_1 \cdot C_2 \cdot \ldots \cdot C_n,$$

(C_i – степень фактора корректировки затрат).

$C_i = 1$ – драйвер затрат не применим.

$C_i > 1$ – драйвер затрат увеличивает затраты.

$C_i < 1$ – драйвер затрат уменьшает затраты.

Факторы корректировки затрат могут сказываться на оценках графика и затрат проекта программной системы, изменяя их в 10 и более раз.

Итак, числовые значения драйверов затрат при их совместном перемножении образуют фактор корректировки, т.е.

$$C = \text{RELY}\times\text{DATA}\times\text{CPLX}\times\text{TIME}\times\text{STOR}\times\text{VIRT}\times\text{TURN}\times\text{ACAP}\times\text{AEXP}\times\text{PCAP}\times \times\text{VEXP}\times\text{LEXP}\times\text{MODP}\times\text{TOOL}\times\text{SCED}.$$

Поскольку драйверы затрат являются мультипликативными, в случае, если драйвер затрат не влияет на трудозатраты, его значение равно 1. При этом конечное значение C не изменяется. Подобные драйверы затрат называются «номинальными» либо «нормальными».

Например, если опыт в области языков программирования (LEXP) команды разработчиков N-вариантной структуры ПО больше, чем аналогичный показатель в любой другой организации, значение LEXP будет оставаться равным 1. Это связано с тем, что превосходящие способности в области языков программирования *нормируются* в данной среде. Оценщик может выполнять поиск условий, при наступлении которых возрастает показатель трудозатрат ($C > 1$) либо значение этого показателя уменьшается ($C < 1$). При поиске применяется критерий «обычности» для данной среды.

Как правило, объем трудозатрат увеличивается, если применяется новая технология, команда разработчиков только что сформирована, либо состоит из неопытных в данной области (например, N-version programming [6]) программистов, имеет место повышенная сложность технологической проблемы (критический по отказоустойчивости объект управления) либо имеют место другие условия, отличные от стандартных. Если же требуется меньше трудозатрат, то это означает, что подобные проблемы были успешно решены ранее.

Следуя идеологии Б.У. Боэма при реализации модели COCOMO мы вынуждены многоатрибутивные оценки драйверов затрат формировать в числовом виде по качественным показателям, отражающим такие понятия, как *«очень низкий»*, *«низкий»*, *«номинальный»*, *«высокий»*, *«очень высокий»* и т.д. Методика промежуточной COCOMO демонстрирует, каким образом каждый из перечисленных выше атрибутов (в числовом выражении) определяется для фиксированного количества приложений программной системы. Немаловажным является логическое обоснование присваиваемых атрибутам значений.

Для N-вариантной модульной структуры программной системы предлагается разбиение на специфические продукты и компоненты этих продуктов. Согласно Б.У. Боэму, подобное разбиение называется трехуровневой иерархией продуктов: система, подсистема, модуль. Верхний уровень (уровень системы) используется для применения самых общих отношений, связанных с проектом N-вариантной системы, таких как номинальные трудозатраты и уравнения графика, а также для применения номинальных трудозатрат на уровне проекта и пофазной разбивки графика.

Предлагается описывать самый нижний уровень, уровень модуля, с помощью показателя KLOC в модуле и драйверов затрат, которые могут варьироваться на этом уровне. При этом второй уровень, уровень подсистемы, описывается с помощью оставшихся драйверов затрат и может отличаться в различных подсистемах. Однако этот уровень не будет изменяться в различных модулях, входящих в состав одной подсистемы.

Для каждого компонента можно отдельно производить анализ драйверов затрат, причем подсистемы и модули наследуют драйверы затрат системы, а именно: RELY, VIRT, TURN, MODP, TOOL и SCED. Модули наследуют драйверы затрат подсистем: DATA, TIME, STOR, ACAP, AEXP. Это поддерживает проявляющуюся в современных программных системах тенденцию применять одни и те же модули внутри подсистемы. Драйверы затрат модуля: KLOC, CPLX, PCAP, VEXP и LEXP. Предлагается использовать дополнительный драйвер AAF – результат адаптации существующих модулей и структур ПО. Это важно, когда разрабатываемые N-вариантные структуры ПО создаются не «с нуля», а являются результатом повторного использования предыдущих версий существующих модулей, что характерно, например, для объектно-ориентированного подхода.

Следует отметить, что затраты на этапе повторного использования, как и при автоматической генерации кода модулей (например, при использовании концепции генетического программирования в мультиверсионности [8]), не всегда равны нулю. Возникают трудозатраты на этапе работы с существующим кодом и при разработке соответствующего интерфейса. Затраты на переписывание системы могут быть меньшими, чем продолжение ее поддержки и сопровождения (по причине энтропии структуры). Однако переписывание старой системы может быть более дорогостоящим, чем создание «нового» N-го варианта системы.

Известно требование, согласно которому точка экономической безубыточности достигается в результате изменения 20%

кода; после прохождения этой точки повторное использование кода не будет эффективным [5].

Заключение

Концепция COTS-сопровождения N-вариантного программирования обеспечивает доступность математических моделей для оценки характеристик версий модулей, что позволяет иметь информацию относительно как надежности программной системы, так и ее стоимости. Кроме того, многоатрибутивный подход при управлении трудозатратами на разработку N-вариантного ПО позволяет учесть, что программные модули разрабатываются полностью независимыми программистами (или группами программистов) с использованием различных инструментальных средств и, как правило, в различных операционных средах.

Многоатрибутивные оценки драйверов затрат используют качественную информацию, исключение которой может привести к неоптимальному (и, возможно, неверному) решению. Таким образом, рассматриваемый подход к решению задачи многоатрибутивного управления с использованием драйверов затрат на разработку программных систем, позволяет проводить проектировщику анализ ситуаций, когда некоторая часть всей доступной ему информации может быть как качественной, так и неполной.

Исследования выполнены в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 гг.

Список литературы

1. Боэм Б. Характеристики качества программного обеспечения / Б. Боэм, Дж. Браун, Х. Каспар, М. Липов, Г. МакЛеод, М. Мерит. – М.: Мир, 1981. – 208 с.

2. Боэм Б.У. Инженерное проектирование программного обеспечения: пер. с англ. – М.: Радио и связь, 1985. – 512 с.

3. Ковалев И.В. Многоатрибутивное формирование оптимальных по составу высоконадежных сложных систем: монография / И.В. Ковалев, О.И. Завьялова, В.Е. Сисько, М.Ю. Царев. – Красноярск: Краснояр. гос. аграр. ун-т, 2009. – 166 с.

4. Ковалев И.В. Анализ драйверов затрат при многоатрибутивном формировании мультиверсионных программных систем / И.В. Ковалев, А.Н. Лайков, В.С. Скориков, С.Н. Гриценко // Приборы и системы. Управление, контроль, диагностика. – 2009. – № 10. – С. 78–83.

5. Орлов С.А. Технологии разработки программного обеспечения. – СПб.: Питер, 2002. – 464 с.

6. Соммервилл И. Инженерия программного обеспечения: пер. с англ. – 6-е изд. – М.: Издательский дом «Вильямс», 2002. – 624 с.

7. Фатрелл, Р.Т. Управление программными проектами: достижение оптимального качества при минимуме затрат: пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 1136 с.

8. Feldt, R. Generating diverse software versions with genetic programming: an experimental study // IEE Proc. Softw. – December 1998. – Vol. 145, № 6. – P. 228–236.

Рецензенты:

Петров М.Н., д.т.н., профессор, профессор кафедры системного анализа и исследования операций ГОУ ВПО «Сибирский государственный аэрокосмический университет», г. Красноярск;

Попов Ф.А., д.т.н., профессор, зам. директора по ИТ, Бийский технологический институт (филиал) ГОУ ВПО «Алтайский государственный технологический университет имени И.И. Ползунова», г. Бийск.

Работа поступила в редакцию 09.03.2011.