УДК 65.012.123

О.Е. Федорович[1], А.В. Бабич[2]

[1] *Национальный аэрокосмический университет имени Н.Е. Жуковского «ХАИ», Харьков*
[2] *Полтавский политехнический колледж НТУ «ХПИ», Полтава*

# МЕТОД ОБЕСПЕЧЕНИЯ КАЧЕСТВА ПРОЕКТИРУЕМОЙ МНОГОУРОВНЕВОЙ ПРОГРАММНОЙ СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ КОМПОНЕНТНОГО И ПРЕЦЕДЕНТНОГО ПОДХОДОВ

*Ставится и решается задача компонентного проектирования многоуровневой программной системы (ПС). Сформулированы требования к командам разработчиков для формирования базовой архитектуры с использованием программных компонент, созданных в ходе реализации предыдущих проектов, что позволяет значительно повысить качество и минимизировать проектные риски. Новые компоненты создаются для обеспечения новой или расширения имеющейся функциональности и могут повысить инновационность и привлекательность программного продукта. Команда, реализующая новый проект формирует архитектуру новой ПС используя базу прецедентов, содержащую лексикографически упорядоченные компоненты из прошлых разработок.*

***Ключевые слова:*** *компонентная архитектура программной системы, командная разработка, P-Modeling Framework, база прецедентов, лексикографическое упорядочивание прецедентов.*

## Введение

Сложность современных программных систем (ПС) обусловлена иерархией в управлении, их многоуровневой компонентной архитектурой, присутствием как «старых», так и «новых» задач, связанных с реинжинирингом, рефакторингом кода и расширением сферы применения ПС, многомерностью – как «горизонтальной», так и «вертикальной». Перечень факторов можно расширить, исходя из других, не упомянутых выше, возможных причин усложнения ПС. Поэтому тема данной публикации, в которой предлагается новый подход к обеспечению качества ПС, основанный на компонентном подходе [1, 2] является актуальной. При использовании компонентного подхода целесообразно выделить несколько команд разработчиков, состав которых может соответствовать рекомендациям MSF [3]. Первая команда формирует архитектуру будущей ПС, с учетом долгосрочной стратегии развития ИТ-инфраструктуры предприятия и гипотетических возможных портфелей заказов. Задача этой команды заключается в создании базовой платформы (БП) программных компонент для различных уровней детализации ПС. Для выполнения конкретных заказов по разработке ПС привлекается другая команда разработчиков, которая, исходя из требований технического задания (ТЗ) или структуры проекта по MSF, формирует многокомпонентный состав архитектуры на базе БП конкретной ПС и осуществляет все проектные работы по ее созданию. Третья команда разработчиков, в соответствии с предложенной нами методикой P-Modeling Framework [4] может быть задействована для верификации модели ПС, созданной второй командой, что позволяет повысить качество конечного продукта и полнее учесть требования к создаваемой ПС.

## Постановка задачи исследования

Современная архитектура программной системы (ПС) содержит большое количество компонент, созданных в ходе предыдущих разработок, и используемых в неизменном виде либо адаптированных под нужды нового проекта. Подход, основанный на использовании позитивного опыта прошлых проектов позволит минимизировать риск, связанный с созданием новых программных компонент и обеспечит значительно меньшие затраты на разработку и сокращение сроков создания новой ПС, а также позволит повысить реализуемость проекта. Такой подход полностью соответствует дисциплине управления проектами MSF, которая предполагает обязательный разбор завершенных проектов, извлечение уроков, и пополнение базы знаний, используемой при реализации новых проектов. Появление новых задач управления и необходимость обеспечения новой функциональности ПС приводит к необходимости создания новых компонент. Поэтому, при создании новых ПС необходимо найти компромисс при выборе состава ПС, в который будут входить как компоненты повторного использования, так и новые программные компоненты. Многолетний опыт по созданию ПС позволяет разработчикам выделить основные типы компонент, присущие каждому уровню ее детализации.

Рассмотрим метод синтеза многоуровневой компонентной архитектуры ПС с использованием теории прецедентов, которую целесообразно использовать для формализации опыта прошлых раз-

© О.Е. Федорович, А.В. Бабич

работок. Совместное применение компонентного и прецедентного подходов в общем случае позволяет сократить сроки создания новых ПС с учетом опыта прошлых разработок.

Компонентный подход будет использован для формирования многоуровневой архитектуры новой ПС с использованием хорошо зарекомендовавших себя программных компонент, созданных в ходе прошлых разработок, представленных в виде прецедентов на различных уровнях иерархии создаваемой многоуровневой базы прецедентов (БПР).

## Решение задачи исследования

Архитектура компонентной ПС будет состоять из трёх типов программных компонент [2]. К ним относятся программные компоненты повторного использования (КПИ) не нуждающиеся в адаптации, КПИ нуждающиеся в адаптации, а также новые компоненты (НК). Кроме того, в многоуровневой архитектуре ПС можно выделить составные компоненты, включающие как КПИ, так и НК.

Каждый прецедент в БПР можно представить в виде программного модуля, который описывается с помощью технических характеристик (ТХ) компонента $r_{ij}$ (i-го уровня, j-го типа) и списка проектных работ, связанных с его созданием. Представим технические характеристики компонента $r_{ij}$ в виде кортежа $Q_{r_{ij}}$, каждый элемент которого соответствует конкретной технической характеристике.

Требования по созданию отдельного программного компонента новой ПС, которые содержатся в ТЗ можно также оформить в виде кортежа технических характеристик $Q_{S_{ij}}$, описывающего проблемную проектную ситуацию, разрешение которой можно осуществить с помощью созданной базы прецедентов (БПР). Путём направленного поиска и сопоставления требований $Q_{S_{ij}}$ и каждого $Q_{r_{ij}}$ из БПР можно найти множество прецедентов в виде КПИ на данном i-м уровне представления архитектуры ПС. Если найденные «близкие» $Q_r$ компоненты на рассматриваемом уровне i не удовлетворяют требования проектировщика, он может перейти к следующему (нижнему) уровню декомпозиции ПС (i + 1) и продолжать поиск «близких» прецедентов.

Таким образом, синтез архитектуры новой ПС представляет собой итерационную процедуру поиска прецедентов в многоуровневой БПР программных компонентов.

Заметим, что на нулевом, самом верхнем, уровне декомпозиции в многоуровневой БПР каждый элемент представляет собой существующую ПС, которая была разработана ранее. На первом уровне декомпозиции БПР прецеденты представлены основными функциональными подсистемами, библиотеками или программными комплексами, которые описываются своими ТХ. В БПР может содержаться также описание проектных работ по созданию данных подсистем или библиотек. В последующих уровнях БПР содержатся КПИ с различной степенью детализации архитектуры ПС.

Пусть для каждого функционального компонента разрабатываемой новой ПС, с учётом i-го уровня декомпозиции архитектуры, наименования j существует множество прецедентов $M_{ij}$ в БПР в виде компонентов, созданных в ходе предыдущих проектов. Это множество прецедентов было предварительно сформировано разработчиками базовой платформы (БП).

Необходимо найти подмножество $M_{ij}^{*}$ в БПР, которое «ближе» всего по техническим характеристикам к необходимому нам программному компоненту ПС (новая проблемная ситуация). Для выполнения поисковых операций в БПР проведём предварительное упорядочивание прецедентов. Это необходимо осуществить на каждом уровне *i* декомпозиции архитектуры ПС для функционального наименования j. Для предварительного ранжирования прецедентов удобно воспользоваться лексикографическим упорядочиванием. Каждый компонент (соответствующий ему прецедент) представим в БПР в виде «слова» (кортежа) технических характеристик. Технические характеристики в «слове» ранжируются в порядке важности.

На первом месте «слова» находится значение наиболее значимой в контексте текущего проекта технической характеристики компонента, а на последнем – наименее значимой. Для обеспечения удобства поиска выполним перевод значений всех технических характеристик программных компонент в качественные значения лингвистических переменных $l_{ijb}$, где i – уровень декомпозиции архитектуры ПС, j – наименование программного компонента, b – техническая характеристика.

Пусть качественное значение лингвистической переменной $l_{ijb}$ соответствует буквам латинского алфавита. Например, A – лучшее, наиболее подходящее значение характеристики; B – отличное; C – хорошее; D – удовлетворительное.

Пусть заданы диапазоны значений для конкретных ТХ программных компонентов, которые однозначно связаны с качественными значениями каждой лингвистической переменной. Кроме того, что осуществляется перевод ТХ компонентов в качественные значения и упорядочение их внутри «слов», исходя из важности конкретных характеристик, лексикографическое упорядочение обеспечивает направленный поиск и сопоставление требований $Q_{s_{ij}}$ с каждым прецедентом $Q_{r_{ij}}$ из БПР.

В результате проведённого лексикографического упорядочения всё множество прецедентов $M_{ij}$

заданного уровня декомпозиции можно представить в виде упорядоченного списка. В упорядоченном списке прецедентов вначале будут находиться те прецеденты, которые являются наиболее «близкими» по своим ТХ к необходимому компоненту $s_{ij}$ новой ПС.

Проектировщик может выбирать определенное количество прецедентов $Q_r$ из этого списка для последующего рассмотрения и оптимизации на своё усмотрение. Отобранные прецеденты из упорядоченного списка можно представить в виде подмножества $M^*_{ij}$.

Пусть, любой $r_{ij}$-й программный компонент (прецедент) в БПР представлен в виде кортежа («слова») $Q_{r_{ij}}$ с элементами в виде значений технических характеристик (на первом месте находится наиболее значимая характеристика, а на последнем – наименее значимая). Например,

$$Q_{r_{ij}} = A_{r_{ij}} C_{r_{ij}} A_{r_{ij}} B_{r_{ij}} \ldots .$$

Лексикографическое упорядочивание прецедентов в БПР и требований к проектируемым компонентам имеет ряд преимуществ. К ним относятся:

– ТХ в «слове» каждого программного компонента ранжируются по важности, что даёт возможность осуществлять поиск прецедентов для требуемых компонент различного уровня иерархии, сосредотачиваясь на наиболее приоритетных характеристиках компонента;

– диапазоны значений или конкретные значения ТХ, сопоставляющие представленные качественные значения (*A, B, C, D, …*) определённым значениям лингвистической переменной позволяют объективно учитывать пожелания заказчика;

– вне зависимости от того, качественными или количественными значениями представлены ТХ прецедентов и требуемых компонент, все ТХ программной компоненты оформляются в виде «слова», состоящего из качественных значений лингвистических переменных, отвечающих за конкретную ТХ, заданного уровня декомпозиции;

– поиск наиболее «близких» компонентов осуществляется на основании сравнения упорядоченных значений каждой из технических характеристик прецедентов и требуемых компонентов нового проекта, что обеспечивает простоту в получении результата поиска.

Следует отметить, что лексикографическое упорядочивание прецедентов в БПР и требований к проектируемым программным компонентам осуществляется на конкретно заданном уровне декомпозиции ПС. Если же проектировщик счёл необходимым рассмотреть ПС «глубже» и поискать программные прецеденты на более нижнем уровне, тогда осуществляется лексикографическое упорядочение БПР и детализация требований к проектируемым программным компонентам на более низком уровне декомпозиции.

При выполнении конкретного проекта основные требования к ПС, сформулированы в техническом задании (ТЗ) или документе структуры проекта. Поэтому для выбора компонент из базы прецедентов (БПР) необходимо учитывать следующие факторы:

- удовлетворение требований к ПС (F1);
- удовлетворение значений технических характеристик (F2);
- удовлетворение требований, касающихся эксплуатации ПС (F3);
- удовлетворение требований масштабируемости ПС (F4).

При выборе компонент для выполнения проекта по созданию новой ПС будем пользоваться множеством КПИ, которые ранее сформированы в виде многоуровневой базы прецедентов (БПР) разработчиками базовой платформы (БП). На примере прокомментируем возможные варианты формирования компонентной архитектуры ПС для нового проекта.

Первый вариант (слева на рис. 1) соответствует выбору в качестве КПИ готовой подсистемы, которую, в дальнейшем, возможно, потребуется адаптировать под требования конкретного заказчика.

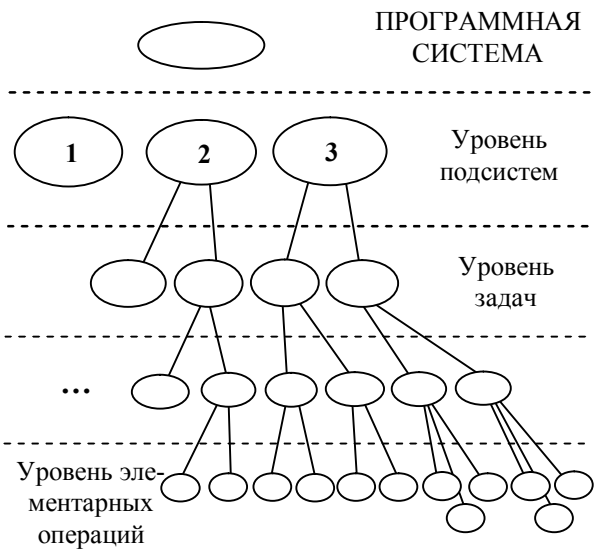


Рис. 1. Иерархия формирования компонентной программной системы

Второй вариант (в середине рис. 1) соответствует формированию архитектуры новой ПС из программных компонентов разных уровней иерархии БПР. При этом возможно появление новых программных компонент, которые, в основном, формируются (разрабатываются) за счет композиции программных компонент нижнего уровня (уровней).

Третий вариант (справа на рис. 1) соответствует проекту по созданию ПС с большой степенью

новизны, когда программные компоненты разных уровней формируются из программных элементов самого нижнего уровня. При этом предполагается, что существует программная элементная база (вплоть до программного языка нижнего уровня) для проектирования новой ПС.

Оценим возможные варианты по уровню инновационности, качеству, затратам и рискам.

Первый вариант разработки ПС будет иметь минимальные затраты, потенциально максимальное качество, минимальную инновационность, риски возникают вследствие интеграции имеющихся КПИ и адаптации к требованиям конкретного заказчика.

Второй вариант наиболее часто встречается в практике проектирования ПС и включает в себя как старые, уже апробированные решения, так и новые, связанные обеспечением новой функциональности ПС. Риски снижения качества обусловлены необходимостью создания новых программных компонент. Этот же фактор влияет на общие проектные риски и, соответственно, реализуемость проекта.

Третий вариант соответствует высокой инновационности, но имеет самые большие проектные риски и затраты, связанные как с реализацией проекта, так и с обеспечением требуемого качества.

Подробнее рассмотрим процесс поиска и выбора требуемых для конкретного проекта компонент повторного использования (КПИ), которые ранее были отобраны для базовой платформы и размещены в многоуровневой базе прецедентов (БПР) по уровням архитектуры гипотетической ПС усилиями команды разработчиков базовой архитектуры (БА).

При позиционировании архитектуры новой разрабатываемой ПС для каждой функциональной компоненты нового заказа $r_{ij}$ (i – уровень, j – функциональность) необходимо найти «близкие», в заданном представлении близости, КПИ. Возможны следующие варианты «близости» $r_{ij}$ по отношению к множеству $M_{ij}$ КПИ для $f_{ij}$ функциональной компоненты в БПР базовой архитектуры ПС.

Пусть для проведения операции поиска «близких» вариантов в БПР использовано лексикографическое упорядочивание вариантов. Упорядочивание проводится с учетом важности показателей вектора

$$F = \{F_1, F_2, F_3, F_4, F_5\},$$

а также с учетом требований качества $Q$, затрат $W$, времени $T$ и рисков $R$ [3].

1. Первый вариант соответствует чрезвычайно высокому значению показателей для $f_{ij}$ функциональной компоненты нового проекта, которые значительно превышают возможные значения показателей для множества существующих КПИ, связанного с $f_{ij}$ в БПР, что нарушает представление «близости» для экспертов (рис. 2).

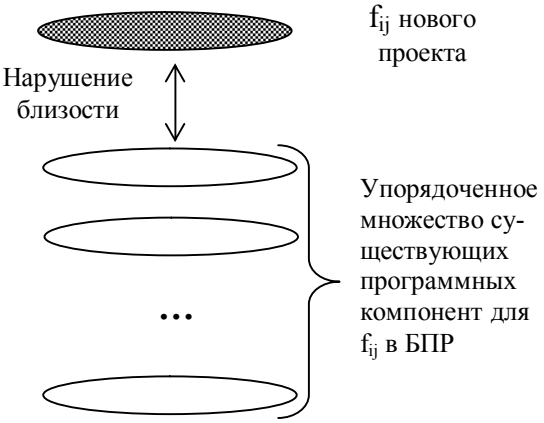


Рис. 2. Значительное превышение значений показателей в новом проекте:
– обозначение значений показателей ТЗ нового проекта

В этом случае, возможны следующие варианты действий разработчиков новой ПС:

– путем поиска новых существующих программных компонент повторного использования расширить множество возможных компонент для $f_{ij}$ функциональной компоненты в БПР;

– смягчить «жесткие» требования по отношению к $f_{ij}$ функциональной компоненте;

– при проектировании новой ПС рассматривать $f_{ij}$ как новую программную компоненту, которую необходимо разработать «с нуля».

2. Второй вариант соответствует низкому значению показателей для разрабатываемой $f_{ij}$ функциональной компоненты (рис. 3).

В этом случае выбор программных компонент повторного использования осуществляется достаточно просто с учетом требований экспертов и разработчиков в множестве КПИ, которое присутствует в БПР.


Рис. 3. Низкие значения показателей проекта

3. Наиболее часто встречающаяся на практике ситуация показаны на рис. 4.

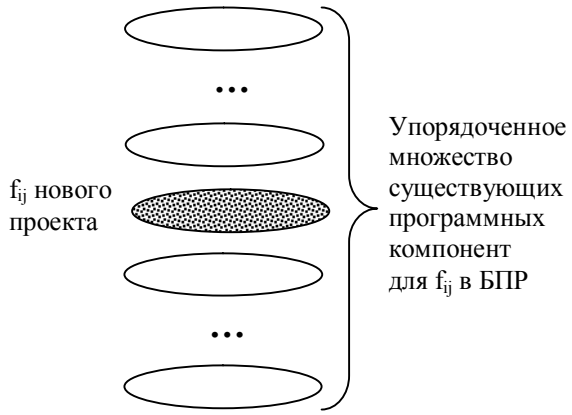


Рис. 4. Реальная ситуация при выполнении нового проекта

В этом случае, с учетом заданного критерия «близости», осуществляется поиск требуемой компоненты в множестве КПИ, которые отобраны и существуют в виде упорядоченного множества в БПР.

## Выводы

Предлагаемый подход целесообразно использовать при выполнении новых проектов по созданию ПС. Для этого, предварительно, с учетом стратегии развития IT-инфраструктуры предприятия, формируется компонентная архитектура гипотетической ПС усилиями специально выделенной команды разработчиков.

Выполнение конкретного проекта осуществляется другой командой, с максимальным использованием программных компонент из прошлых разработок, что облегчает работу проектной команды, уменьшает риски, сокращает сроки и затраты разработки. Верификация модели ПС осуществляется отдельной командой проектировщиков, что позволяет повысить качество конечного продукта и полнее учесть требования к создаваемой ПС. Накопление информации о программных компонентах, используемых при построении многоуровневой базовой архитектуры, а также о проектных задачах, которые необходимо выполнить для разработки или использования каждой компоненты, осуществляется с использованием базы прецедентов.

## Список литературы

1. *Brown A. Large Scale, Component-Based Development [Text] / A. Brown. – Prentice Hall, 2000. – 285 p.*

2. *Федорович О.Е. Компонентное проектирование аэрокосмической техники [Текст] : моногр. / О.Е. Федорович, Е. С. Яшина, И. В. Белецкий. – Х. : Нац. аэрокосм. ун-т «Харьк. авиац. ин-т», 2012. – 180 с.*

3. *Using Reverse Semantic Traceability for Quality Control in Agile MSF-based Projects [Electronic resource] / K. Zhereb, V. Pavlov, A. Doroshenko, V. Sergienko // International Software & Productivity Engineering Institute (INTSPEI). – Moscow, Russia: 4th Software Engineering Conference, October 23, 2008: Proc. – Mode of access: WWW. URL: http://2008.cee-secr.org/ru/etc/secr2008_ konstantin_zhereb_intspei_using_rst.pdf. – Last access: March 11, 2016. – Title from the screen.*

4. *Applying Pantomime and Reverse Engineering Techniques in Software Engineering Education [Text] / V.L. Pavlov, N. Boyko, A. Babich, O. Kuchaiev, S. Busygin // 37th ASEE/IEEE Frontiers in Education Conference. – Milwaukee, Wisconsin, USA, 2007. – P. T1E-1- T1E-5.*

*Поступила в редколлегию 11.04.2016*

**Рецензент:** д-р техн. наук, проф. И.В. Шостак, Национальный аэрокосмический университет им. Н. Е. Жуковского «ХАИ», Харьков.

### МЕТОД ЗАБЕЗПЕЧЕННЯ ЯКОСТІ ПРОЕКТОВАНОЇ БАГАТОРІВНЕВОЇ ПРОГРАМНОЇ СИСТЕМИ З ВИКОРИСТАННЯМ КОМПОНЕНТНОГО ТА ПРЕЦЕДЕНТНОГО ПІДХОДІВ

О.Є. Федорович, О.В. Бабич

*Ставиться й вирішується завдання компонентного проектування багаторівневої програмної системи (ПС). Сформульовано вимоги до команд розробників для формування базової архітектури з використанням програмних компонент, створених в ході реалізації попередніх проектів, що дозволяє значно підвищити якість і мінімізувати проектні ризики. Нові компоненти створюються для забезпечення нової або розширення наявної функціональності і можуть по-висіть інноваційність і привабливість програмного продукту. Команда, яка реалізує новий проект формує архітектуру нової ПС використовуючи базу прецедентів, що містить лексикографічно впорядковані компоненти з минулих розробок.*

***Ключові слова:*** *компонентна архітектура програмної системи, база прецедентів, командна розробка, P-Modeling Framework, лексикографічне впорядкування прецедентів.*

### QUALITY ASSURANCE METHOD TO DESIGN A MULTI-LEVEL SOFTWARE SYSTEM USING THE COMPONENT-BASED AND USE-CASE APPROACHES

O.Ye. Fedorovych, A.V. Babich

*The problem of designing a multi-level component of a software system (PS) is posed and solved. The requirements to the development teams are formulated to form the basic architecture with the use of software components created during the implementation of previous projects, which ensures quality and minimize project risks. The new components are created to provide a new or expand existing functionality and can improve the innovativeness and attractiveness of the product. The team creates a new architecture using precedent database that contains the lexicographically ordered components from past designs.*

***Keywords:*** *component architecture of a software system, precedents base, team development, P-Modeling Framework, lexicographically ordered precedents.*