Международный журнал прикладных наук и технологий "Integral"

Научная статья

Original article

УДК 004.41

DOI 10.55186/27131424_2023_5_2_4

ЭВОЛЮЦИЯ РАЗВИТИЯ ЭМПИРИЧЕСКОЙ ИНЖЕНЕРИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

THE EVOLUTION OF EMPIRICAL SOFTWARE ENGINEERING

Бузыкова Юлия Сергеевна, доцент кафедры математического обеспечения и стандартизации информационных технологий, Федеральное государственное бюджетное образовательное учреждение высшего образования "МИРЭА - Российский технологический университет" (119296 Россия, г. Москва, ул. проспект Вернадского, д. 78, стр. 4), тел. +7(495)506-44-55, juliaserg_buz@mail.ru

Зуфарова Анна Сергеевна, старший преподавать кафедры «Высшая математика», Тихоокеанский государственный университет (680035 Россия, г. Хабаровск, ул. Тихоокеанская, д. 136), тел. +7(495)588-22-47, zoof_anna@mail.ru

Yulia S. Buzykova, Associate Professor of the Department of Mathematical Support and Standardization of Information Technologies, Federal State Budgetary Educational Institution of Higher Education "MIREA - Russian Technological University" (78 prospekt Vernadskogo st., p. 4, Moscow, 119296 Russia), tel. +7(495)506-44-55, juliaserg_buz@mail.ru

Anna S. Zufarova, Senior lecturer of the Department of "Higher Mathematics", Pacific NationalUniversity (136 Pacific Street, Khabarovsk, 680035 Russia), tel. +7(495)588-22-47, zoof_anna@mail.ru

Аннотация. Эволюция программного обеспечения (ПО) означает динамическое поведение программных систем, когда они поддерживаются и развиваются на протяжении всей жизни. Эволюция по особенно важна, поскольку за счет именно возможности эволюционировать системы становятся все более долговечными.

Сложность эмпирической инженерии программного обеспечения (ЭИПО) так как и проведение эмпирических исследований в других областях знаний связана со сбором информации об объекте исследования, выполнении эксперимента и выдвижении гипотез.

Эмпирическая Инженерия программного обеспечения используется для ответа на эмпирические вопросы относительно ПО, которое разрабатывается. При исследовании формируются теории, предположения и гипотезы. Из этих гипотез делаются прогнозирования конкретных событий.

Предсказания проверяются соответствующими экспериментами. В зависимости от результатов эксперимента формулировки подтверждаются или опровергаются.

Эмпирические исследования дают эмпирические данные, которые затем могут быть проанализированы на статистическую значимость для дальнейшего принятия решений.

Abstract. Software evolution refers to the dynamic behavior of software systems when they are maintained and developed throughout their lives. The evolution of software is especially important, because due to the ability to evolve, systems are becoming more and more durable.

The complexity of empirical software engineering (EIPO), as well as conducting empirical research in other fields of knowledge, is associated with collecting information about the object of research, performing an experiment and putting forward hypotheses.

Empirical Software Engineering is used to answer empirical questions about software that is being developed. Theories, assumptions and hypotheses are formed during the research. Predictions of specific events are made from these hypotheses.

Predictions are verified by appropriate experiments. Depending on the results of the experiment, the formulations are confirmed or refuted.

Empirical studies provide empirical data that can then be analyzed for statistical significance for further decision-making.

Ключевые слова: *эволюция, эмпирическая инженерия, программное обеспечение, исследование*

Keywords: *evolution, empirical engineering, software, research*

Эмпирическая инженерия программного обеспечения — совокупность действий для получения знаний с целью лучшего понимания аспектов разработки программного обеспечения. Результатом действий является ряд утверждений относительно определенного перечня проблем. Эти утверждения являются ответами на поставленные вопросы и подтверждением или опровержением гипотез. Эволюционные явления в программных областях не ограничиваются программами и связанными артефактами, техническими характеристиками, проектами и документацией.

Применения, определения носителя, цели, парадигмы, алгоритмы, языки, практики использования, подпроцессы и процессы разработки ПО и т. п — также являются эволюционными явлениями.

Эти эволюционирующие субъекты взаимодействуют и влияют друг на друга. Если их эволюция должна быть дисциплинированной, то соответствующая эволюция процессов должна быть запланирована и управляема. Чтобы они были освоенными, они должны быть понятными и освоенными индивидуально и совместно [6].

Признание эволюции ПО, его идентификация как дисциплинированное явление и его дальнейшее изучение было вызвано докладом 1968-1969 гг., что имеет название «Процесс программирования» [6].

В частности, в исследовании рассмотрены эмпирические данные о росте операционной системы IBM OS/360-370. Был сделан вывод, что эволюция системы измеряется ростом размера над последовательными релизами, где отражается регулярность, которая вряд ли была в первую очередь решением человека [4].

М. М. Леман выделил следующие типы программ: S — программы написаны в строгом соответствии со спецификацией того, что программа может делать; P —

программы, которые реализуют процедуры, которые полностью определяют их поведение; E — программы, осуществляющие работу в условиях реального мира, то есть существенно зависимы от среды своего функционирования, а потому нуждаются в адаптации к тем или иным внешним требованиям [6].

Еще в 1970-х годах Н. М. Лемман начал формулировать законы эволюции программного обеспечения, осознав необходимость разработки программных систем [6]. При этом выделил законы Е-типа.

Эти законы можно разделить на три категории:

I. законы об эволюции характеристик программных систем;

II. Законы, касающиеся организационно-экономических ограничений на развитие ПО; III. Цель-законы эволюции ПО.

Первое систематическое изучение законов Белади и Лемана было выполнено одним из студентов Лемана Чонг Хок Юэном. Его работа была представлена в серии из трех эмпирических работ, опубликованные в 1985, 1987 и 1988 годах [5]. Автор сообщает о результате, где применяется дефект обнаружения и коррекции, включая наблюдение. Время, которое было выделено для исправления дефекта, не увеличивался, как можно было ожидать из-за роста сложности системы. Автор повторно анализирует три различных системы и несколько других систем, и смотрит на различные зависимые переменные такие как: количество модулей, процент обработанных данных, продолжительность фазы .

После просмотра данных из предыдущих исследований характерные для ОС/360 не обязательно исследовать другие системы; первые два закона были подтверждены, остальные законы не были исследованы. Однако автор отмечает, что последние законы больше основаны на человеческих организациях, занимающихся процессом технического обслуживания чем свойства самого ПО. Юэн продолжает свою экспертизу данных В-систем проводился анализ временных рядов на количество ошибок.

Автор отмечает, что данные которые изучались на предыдущих этапах, он называет их «глобальными» данным обслуживанию (наблюдаются на глобальном уровне), как правило, демонстрируют мало результатов. Эти результаты ранее были

интерпретированы как инвариация и определены средние их значения, которые привели к эволюционной динамике.

Рассматривается техническое обслуживание большой части ПО на уровне «О», а также на глобальном уровне. Помимо тестов автор также использует Time Series Analysis-методы спектрального анализа (коррелограма, хи-квадрат, авторегрессия), а также линейную фильтрацию (подвижные средние методы). Автор сообщает о разные результаты, в том числе о тех, где внешние факторы играют большую роль в определении амплитуды каждого пика и интервал между последовательными вершинами [1].

В Японии использовали опросник — обзор японских организаций для изучения бизнеса. Замена по происходит в течение 5 лет. Они подают ряд описательных результатов: по меньшему масштабу имеет тенденцию иметь более короткий срок службы, административный тип программ, как правило, имеют более длительный срок службы, чем поддержка типовых программ (например, продажная поддержка, производство) [4].

Другие ученые занимались поиском данных при факторном анализе программных показателей. Авторы делают вывод, что информационные показатели, такие как методы Halstead и McCabe, а также простые методы анализа достоверны. Они также рассматривали эволюцию системы в течение 18 месяцев и утверждали, что нашли доказательства законов эволюции, ссылаясь на постоянные изменения, растущую энтропию [2].

Ученые исследовали две распределительные модели модификаций технического обслуживания, чтобы определить, поддерживает ли по однородное распределение. Авторы изучали отчеты о проблемах ПО (SPR) из системы 4GL. Эти SPR характеризуются модификационным типом (корректирующие или адаптивные). Кроме того, они отслеживают количество модификаций, вызванных предыдущими. Авторы отметили, что уровень поддержки модификации уменьшаются во времени, но если рассматривать в отдельных фазах, которые они описывают как «стабилизация», «совершенствование», и «расширение», то наоборот увеличивается [3].

В процессе исследования определялся программный код, который является пригодным для повторного использования, с помощью статистики и проведения расчетов, на основе проведенного анализа метрик 100 ПО [1]. Предлагалось автоматизировать процесс оценки путем определения прямых метрик, которые получаются в результате измерения программного обеспечения, на косвенные метрики, которые оценивают эксперты. Все метрики (прямые и косвенные) были выбраны из расчета возможности их использования для определения повторно используемого программного кода.

Предложен Метод определения зависимостей между метриками программного обеспечения с помощью статистического анализа».

В общем виде статистический анализ, который выполняется с целью определения зависимостей, состоит из трех этапов: первичный статистический анализ, корреляционный анализ и регрессионный анализ [3].

Этот метод характеризуется следующим образом:

- отсутствие определения закона распределения метрики;
- обязательное большое значение выборки; использование расчета только парной ранговой корреляции;
- отсутствие проверки точности коэффициентов корреляции;
- отсутствие проверки общего закона распределения метрик;
- построение регрессии методом линеаризации.

Ученые исследовали различные метрики на этапе проектирования для того, чтобы уже на этом этапе узнать сложность системы. Сложность системы является качественной характеристикой. Уменьшение сложности ПС позволяет снизить трудоемкость проектирования, разработки, тестирования и сопровождения, обеспечивает простоту и надежность производимой ПС [4].

На этапе проектирования оценивания сложности осуществляется оценке отдельных компонентов системы и связей между ними. Связность и сцепление классов имеют аналогию со связностью и сцеплением модулей. Классы соответствуют модулям, а функции — методам. Увеличение внутренней связности

и уменьшение внешнего сцепления снижает сложность и способствует обеспечению надежности программных систем [2].

К основным результатам можно отнести постепенное ознакомление с развитием эмпирических методов, отношения исследователей к работам других ученых и анализ основных законов эмпирической инженерии программного обеспечения и их методов.

Выводы

Проведенный анализ развития эмпирической программной инженерии, исследованы работы других исследователей, систематизированы и обобщены данные опытов в виде таблиц. Подводя итоги можно сказать, что для каждого применения эмпирического метода нужно определиться с целью, то есть проблемой опыта, использовать различные подходы к решению определенной задачи и чем больше экспериментов, опытов и практик тем лучше. Каждый метод должен применяться в своей сфере. В эмпирической инженерии программного обеспечения объектом опыта является программа, и цель опытов выявить износ, конкурентоспособность, экономическую справедливость и время актуальности данной системы.

Литература

1. Акбаров Х.У. Математическая модель погрешностей обработки на прецизионных токарных станках с чпу // Universum: технические науки. 2020. №11-1 (80). С. 101113.
2. Аль-Обайди Луаи М. Р., Попов М. Е. Повышение производительности обработки валов на токарном станке с ЧПУ // Молодой исследователь Дона. 2019. №2 (17). С. 150-158.
3. Маткаримов, Б.Б. Точности обработки на станках с чпу // ORIENSS. 2021. №11. С. 89-96.
4. Насибуллин А.А. Управление рисками в условия интеллектуализации цифровых таможенных технологий // Вестник Российской таможенной академии. 2021 г. № 1. С. 153-159.

5. Онтологическое моделирование предприятий: методы и технологии: моногр. / С. В. Горшков, С. С. Кралин, О. И. Муштак и др.; отв. ред. С. В. Горшков. - Екатеринбург: Издательство Уральского университета. 2019. С. 234.
6. Стрельников Р. В. SOC. Неэффективность внедрения // Вестник Балтийского федерального университета им. И. Канта. Сер.: Физико-математические и технические науки. 2019. № 4. С. 81 - 85.
7. Чучалин А.И. Адаптация "The Core CDIO Standards 3.0" к высшему STEM-образованию // Высшее образование в России. 2021. Т. 30. № 2. С. 9-21. DOI: 10.31992/0869-3617-202130-2-9-21
8. Kotis K., Vouros G., & Spiliotopoulos D. Ontology engineering methodologies for the evolution of living and reused ontologies: Status, trends, findings and recommendations // The Knowledge Engineering Review. 2020. Vol. 35. E4. doi:10.1017/S0269888920000065.

References

1. Акбаров Х.У. Математическая модель погрешностей обработки на прецизионных токарных станках с чпу // Universum: технические науки. 2020. №11-1 (80). С. 101113.
2. Аль-Обайди Луаи М. Р., Попов М. Е. Повышение производительности обработки валов на токарном станке с ЧПУ // Молодой исследователь Дона. 2019. №2 (17). С. 150-158.
3. Маткаримов, Б.Б. Точности обработки на станках с чпу // ОРИЕНСС. 2021. №11. С. 89-96.
4. Насибуллин А.А. Управление рисками в условия интеллектуализации цифровых таможенных технологий // Вестник Российской таможенной академии. 2021 г. № 1. С. 153-159.
5. Онтологическое моделирование предприятий: методы и технологии: моногр. / С. В. Горшков, С. С. Кралин, О. И. Муштак и др.; отв. ред. С. В. Горшков. - Екатеринбург: Издательство Уральского университета. 2019. С. 234.

6. Стрельников Р. В. СОК. Неэффективность внедрения // Вестник Балтийского федерального университета им. И. Канта. Сер.: Физико-математические и технические науки. 2019. № 4. С. 81 - 85.
7. Чучалин А.И. Адаптация "основных стандартов CDIO 3.0" к высшему стволовых образованию // Высшее образование в России. 2021. Т. 30. № 2. С. 9-21. DOI: 10.31992/0869-3617-202130-2-9-21
8. Котис К., Вурос Г. и Спилиотопулос Д. Методологии разработки онтологий для эволюции живых и повторно используемых онтологий: статус, тенденции, выводы и рекомендации // Обзор разработки знаний. 2020. Том 35. E4. doi:10.1017/S0269888920000065.

Для цитирования: Бузыкова Ю.С., Зуфарова А.С. ЭВОЛЮЦИЯ РАЗВИТИЯ ЭМПИРИЧЕСКОЙ ИНЖЕНЕРИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ // Международный журнал прикладных наук и технологий "Integral" №2/2023