УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ноябрь–декабрь 2019 Том 19 № 6 ISSN 2226-1494 http://ntv.itmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTCS
November–December 2019 Vol. 19 No 6 ISSN 2226-1494 http://ntv.itmo.ru/en/
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК УНИВЕРСИТЕТ ИТМО ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

УДК 004.4’23 doi: 10.17586/2226-1494-2019-19-6-1079-1085

АРХИТЕКТУРА ИНТЕГРИРОВАННОЙ СРЕДЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ПОДДЕРЖКОЙ СТРУКТУРНОГО РЕДАКТИРОВАНИЯ

Н.В. Ванясин[a], И.Г. Сидоркина[a], В.И. Поляков[b]

[a] Поволжский государственный технологический университет, Йошкар-Ола, 424000, Российская Федерация
[b] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
Адрес для переписки: SidorkinaIG@volgatech.net

Информация о статье
Поступила в редакцию 02.09.19, принята к печати 10.10.19
Язык статьи — русский

Ссылка для цитирования: Ванясин Н.В., Сидоркина И.Г., Поляков В.И. Архитектура интегрированной среды разработки программного обеспечения с поддержкой структурного редактирования // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 6. С. 1079–1085. doi: 10.17586/2226-1494-2019-19-6-1079-1085

Аннотация

Предмет исследования. Представлены результаты исследования инструментальных средств разработки программного обеспечения. Решена задача создания архитектуры среды разработки программного обеспечения с поддержкой структурного редактирования. Показан процесс проектирования и экспериментальные проверки модифицированной архитектуры интегрированной среды разработки. **Метод.** На основе анализа существующих интегрированных сред разработки программного обеспечения с поддержкой структурного редактирования выделены их особенности и недостатки. Предложена архитектура среды разработки, позволяющая создать прототип системы для решения таких проблем существующих структурных редакторов, как круговая трансляция из текстового представления исходного кода в промежуточное и обратно, недостаточный функционал (по сравнению с классическими средами разработки), сложность разработки новых сред с поддержкой структурного редактирования. **Основные результаты.** При проектировании архитектуры предложено отказаться от текстового представления исходного кода на всех этапах разработки, и это позволило отказаться от круговой трансляции и уменьшить количество компонентов системы. Такое решение привело к повышению производительности среды разработки. Предложенный прототип среды разработки с поддержкой структурного редактирования позволил экспериментальным образом проверить эффективность модификации архитектуры. Выполнены эксперименты по сравнению скорости компиляции программных проектов и скорости проведения автоматического рефакторинга исходного кода. **Практическая значимость.** Предложенное решение может быть использовано в организациях, занимающихся промышленной разработкой программного обеспечения, а также создателями новых сред разработки программного обеспечения.

Ключевые слова
структурное редактирование, среда разработки, интегрированные среды разработки, разработка программного обеспечения, синтаксически-ориентированные редакторы, промежуточное представление кода, оптимизация производительности

Благодарности
Работа выполнена при финансовой поддержке РФФИ (проект № 17-07-00700/19).

doi: 10.17586/2226-1494-2019-19-6-1079-1085

INTEGRATED ENVIRONMENT ARCHITECTURE FOR SOFTWARE DEVELOPMENT WITH STRUCTURED EDITING SUPPORT

N.V. Vanyasin[a], I.G. Sidorkina[a], V.I. Polyakov[b]

[a] Volga State University of Technology, Yoshkar-Ola, 42400, Russian Federation
[b] ITMO University, Saint Petersburg, 197101, Russian Federation
Corresponding author: SidorkinaIG@volgatech.net

Article info
Received 02.09.19, accepted 10.10.19
Article in Russian

For citation: Vanyasin N.V., Sidorkina I.G., Polyakov V.I. Integrated environment architecture for software development with structured editing support. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2019, vol. 19, no. 6, pp. 1079–1085 (in Russian). doi: 10.17586/2226-1494-2019-19-6-1079-1085

Abstract

Subject of Research. The paper presents research of software development tools. The study analyzes existing integrated software development environments with structured editing support. We consider the design process and experimental testing of the modified integrated development environment architecture. **Method.** Analysis of the existing integrated software development environments with structured editing support was performed and their shortcomings were identified. A development environment architecture was proposed. It provides the development of system prototype for such problems as circular translation from textual representation of source code to intermediate one and vice versa, the lack of functionality (compared to classical development environments), design complexity of new development environments with structured editing support. **Main Results.** In this architecture design we have proposed to abandon the textual representation of the source code at all stages of software development. As a result, the circular translation has been eliminated, the number of system components has been reduced, and development environment performance has been increased. The created prototype of development environment with support for structured editing has tested experimentally the effectiveness of the proposed architecture modification. Experiments have been carried out including comparison of compilation speed of software projects and the speed of automatic source code refactoring. **Practical Relevance.** The proposed solution can be used by organizations engaged in the industrial software development and by creators of new software development environments.

Keywords

structured editing, development environments, integrated development environments, software development, syntax-oriented editors, intermediate source code representation, performance optimization

Acknowledgements

This work was supported by the RFBR Grant No. 17-07-00700/19.

Введение

Для разработчиков программного обеспечения (ПО) важна высокая производительность среды разработки при создании программных проектов. Производительность среды разработки может быть оценена такими параметрами, как скорость отклика пользовательского интерфейса среды разработки, скорость компиляции исходного кода, а также скоростью осуществления таких операций, как поиск символа, рефакторинг и т. д.

Разработчики ПО часто используют интегрированные среды разработки (ИСР), которые сочетают в себе все необходимые инструменты для создания ПО. Процесс создания артефактов программного обеспечения представляет собой изменение файлов исходного кода и последующее преобразование этих файлов в набор инструкций для компьютера. Преобразование (компиляция) выполняется достаточно быстро в автоматическом режиме, в то время как внесение изменений в исходный код требует предварительного анализа уже написанного кода, планирование изменений, внесение изменений таким образом, чтобы исходный код оставался семантически верным.

Одним из способов, позволяющих ускорить анализ кода, планирование и внесение изменений, является использование структурных редакторов исходного кода [1]. Структурные редакторы позволяют редактировать не символы и строки текста исходного кода, а структуру программы напрямую. Также известны семантические [2], проекционные [3] и синтаксически-ориентированные редакторы [4].

Однако реализации структурных редакторов часто не используют все преимущества, которые произрастают из структурного подхода к редактированию кода. Например, так как взаимодействие с компилятором происходит только через хранимое представление в виде файлов исходного кода, возникает необходимость круговой трансляции: из промежуточного представления (используемого в редакторе) в текстовое (используемое компилятором и другими внешними инструментами). Это приводит к тому, что реализация не так эффективна в плане оптимизации производительности, как могла бы быть. Еще одной проблемой является небольшой набор функциональных возможностей структурных редакторов по сравнению с классическим подходом к редактированию. Часто отсутствует интеграция с отладчиком, методы автоматического рефакторинга кода и т. д. Это является одной из причин, почему структурные редакторы и среды разработки со структурными редакторами не получили широкого распространения в промышленной разработке ПО [5]. Кроме того, для создания структурного редактора необходима библиотека графических компонентов. В процессе создания редактора должно быть решено множество сложных задач: разработка внешнего вида пользовательского интерфейса и поведения графических компонентов, разработка быстрых алгоритмов отрисовки, определение удобных сочетаний клавиш для навигации, разработка транзакционного поведения для функционала Undo/Redo и др.

Проанализированы существующие среды разработки с поддержкой структурного редактирования: Intentional programming[1] [6], JetBrains MPS[2], Synthesizer generator [7], ProgramTree[3], LavaPE[4], Eclipse

[1] Intentional Software, http://www.intentsoft.com/
[2] JetBrains MPS, http://www.jetbrains.com/mps/
[3] ProgramTree, http://programtree.com/
[4] LavaPE, http://lavape.sourceforge.net/

MultiView plugin[1], iiiiioiooooo[2], Semantic IDE [8]. Анализ показал, что основным недостатком существующих систем является то, что ни в одной из них не представлен весь набор функциональных возможностей, которые доступны в классических средах разработки, таких как Visual Studio, VSCode, JetBrains IDEA и т. д. Также стоит отметить, что все они обладают общим качеством — хранимое представление кода реализовано в виде файлов исходного кода на языке программирования. Это приводит к необходимости постоянной трансляции из текстового представления в промежуточное (для анализа кода, предоставления умных подсказок и т. д.), а затем обратно в текстовое для хранения на накопителе информации и передачи на вход компилятору. Хранение исходного кода в виде уже разобранного промежуточного представления является более оптимальным, так как нивелируется необходимость производить разбор исходного кода при каждой загрузке/компиляции программы [9].

Постановка задачи

Целью настоящего исследования является нахождение такой архитектуры ИСР с поддержкой структурного редактирования, которая позволит минимизировать время выполнения таких операций, как компиляция программы, рефакторинг, отклик пользовательского интерфейса.

Пусть исходный код программного продукта имеет количество строк, равное s, и количество файлов, равное f. Тогда можно задать функцию:

$$T_{\text{общ}}(s, f) = \sum_{k=1}^{3} T_k(s, f),$$

где T_1 — время компиляции программы, T_2 — время выполнения рефакторинга, T_3 — время отклика пользовательского интерфейса.

Таким образом, решение задачи заключается в оценке значения функции при различных значениях переменных s и f для разных архитектур сред разработки.

Архитектура интегрированной среды разработки со структурным редактором

Для решения задачи проектирования среды разработки с поддержкой структурного редактирования предлагается использование промежуточного состояния модели исходного кода на всех этапах работы в среде разработки: для хранения на носителе информации, для передачи информации между модулями среды разработки, а также для компиляции и при отладке программ.

В ходе анализа [10] существующих сред разработки было выявлено, что при классическом подходе к редактированию кода существует необходимость постоянного преобразования текстового представления исходного кода в абстрактное синтаксическое дерево (АСД) для последующих действий над кодом, а также обратное преобразование для сохранения исходного кода в виде текста на накопитель информации. Текстовое представление исходного кода на накопителе информации имеет только одно существенное преимущество: можно использовать классические системы контроля версий для сохранения истории изменений кода [11, 12].

Однако, если отказаться от такого представления исходного кода и всегда использовать промежуточное представление кода (также известное как АСД, CodeDOM или Code Model), то можно повысить производительность среды разработки, так как уменьшится количество шагов в основных алгоритмах взаимодействия между модулями. Предложенная архитектура позволяет упростить подготовку сред разработки для новых языков программирования и их развитие, так как это связано с уменьшением количества компонентов системы.

Предложенная архитектура ИСР с поддержкой структурного редактирования представлена на рис. 1 (описание внутренней архитектуры) и рис. 2 (описание пользовательского интерфейса).

Рассмотрим основные компоненты внутренней архитектуры на рис. 1:

— Core содержит основную логику ПО и управляет промежуточным представлением кода (CodeDOM либо АСД);

— Logger — модуль журналирования действий над исходным кодом ПО. Используется для реализации функции Undo/Redo, а также для формирования дельт для системы контроля версий;

— VCS — система контроля версий;

— Storage — модуль для управления хранимым представлением исходного кода ПО. Может сериализовывать CodeDOM в какой-либо формат хранения и сохранять на накопительное устройство [13];

— Compiler — модуль для компиляции программного проекта. Принимает промежуточное представление кода (CodeDOM или АСД), формирует исполняемые файлы;

— Debugger — модуль отладки. Использует готовые исполняемые файлы и информацию, полученную от других модулей для представления интерфейса отладки;

[1] Eclipse MultiView plugin, http://multiview.cs.pdx.edu/refactoring/statement_view/

[2] iiiiioiooooo, http://celeriac.net/iiiiioiooooo/

— Indexer — модуль для работы с базой знаний об исходном коде ПО. Позволяет находить любые элементы исходного кода по имени или типу элемента. Позволяет реализовать такие функции ИСР, как навигация по коду, рефакторинг, переход к определению и т. д. База знаний может храниться в виде кэша на диске для быстрой повторной загрузки;

— Parser — модуль для работы с уже существующими библиотеками исходного кода ПО. Позволяет разбирать исходный код для использования в модуле Indexer;

— Message Hub — модуль, предоставляющий интерфейс для взаимодействия с пользовательским интерфейсом ИСР. Ограничение взаимодействия через одну точку-интерфейс позволяет сократить время отклика пользовательского интерфейса до минимума;

— Task Manager — модуль для контроля и получения статуса выполнения фоновых задач;

— DI Loader — модуль загрузки ИСР, основанный на шаблоне проектирования «инъекция зависимостей» (Dependency Injection). Позволяет реализовать систему плагинов для расширения существующей архитектуры ИСР. DI позволяет разработчику ИСР заменить любой из компонентов системы своей реализацией [14].

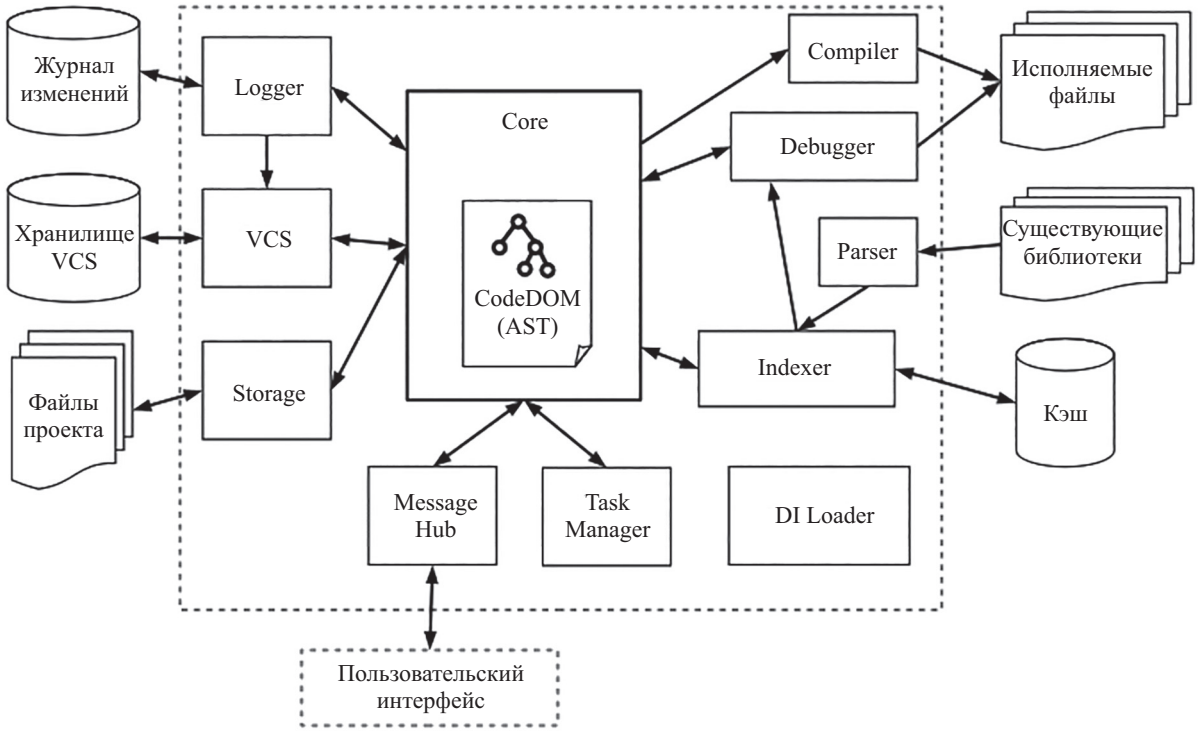

Рис. 1. Архитектура интегрированной среды разработки со структурным редактором (внутренняя часть)

Все изменения промежуточного представления исходного кода контролируются модулем «Controller», обеспечивая целостность данных. Система плагинов (реализуется модулем DI Loader) позволяет расширять систему или добавлять поддержку новых языков программирования. На всех этапах взаимодействия между компонентами системы используется промежуточное представление кода (CodeDOM), что позволяет ускорить взаимодействие между модулями. Промежуточное представление исходного кода является графом языковых конструкций исходного кода [15]. Использован алгоритм, описанный в [16]. Благодаря наличию модуля Parser, который позволяет транслировать исходный код любых внешних библиотек из текстового вида в промежуточное представление кода, обеспечивается возможность работы с уже существующим исходным кодом в текстовом виде. При создании данной архитектуры были использованы паттерны проектирования из [17].

Рассмотрим основные компоненты пользовательского интерфейса (рис. 2):

— Message Hub — модуль, предоставляющий интерфейс для взаимодействия с внутренней частью ИСР (backend);

— DI Loader — модуль аналогичен описанному выше, выполняет те же функции, но для пользовательского интерфейса;

— Controller — управляет промежуточным представлением кода (CodeDOM, либо АСД);

— Editor, Project View, Debugger View — и другие компоненты для отображения представления пользовательского интерфейса (для редактора, обзора структуры проекта, отладчика и т. д.);

— Window Manager — модуль для управления дочерними окнами ИСР;

— Command Executor — модуль для выполнения действий, вызванных пользователем через главное меню или контекстные меню дочерних компонентов.

При проектировании данной архитектуры были использованы принципы построения пользовательского интерфейса для структурных редакторов [18].

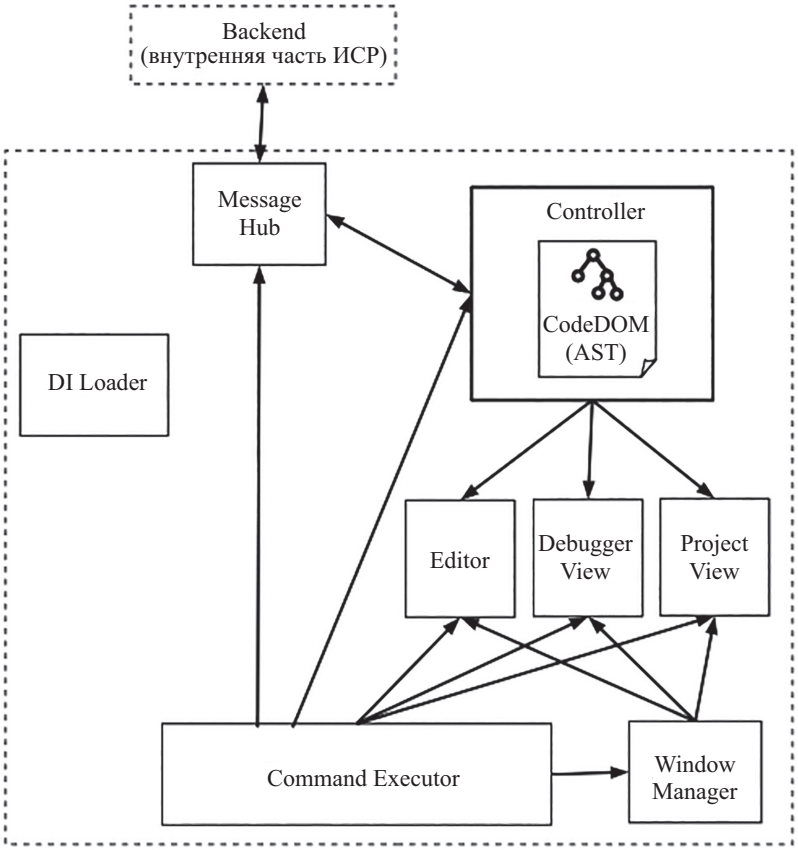

Рис. 2. Архитектура интегрированной среды разработки со структурным редактором (пользовательский интерфейс)

Таким образом, была решена задача создания архитектуры среды разработки со структурным редактированием, которая может позволить увеличить производительность промышленной разработки ПО. Для апробации данной архитектуры необходимо создать опытный прототип – ИСР на основе предложенной архитектуры.

Реализация ИСР со структурным редактором на основе разработанной архитектуры

Разработан программный комплекс — «Интегрированная среда разработки ПО со структурным редактором для языка Go». Сокращенное кодовое название: gostr (Go Structured). Разработка выполнена при помощи языков программирования Go, TypeScript и технологии Electron.

Программный комплекс включает в себя три основных функциональных блока:

1) внутренний функциональный блок среды разработки — backend. Реализует всю логику работы с исходным кодом ПО, интерфейс для компилятора и отладчика, систему контроля версий и другие функции интегрированной среды разработки;

2) модуль для компилятора и отладчика языка Go, позволяющий компилятору использовать промежуточное представление (CodeDOM) исходного кода в качестве входных данных;

3) пользовательский интерфейс для взаимодействия пользователя со средой разработки — frontend. Предоставляет структурный редактор для исходного кода и широкий набор функциональных возможностей для разработки ПО: отладчик, средства рефакторинга, интерфейс для системы контроля версий, навигацию по исходному коду проекта, быстрый поиск, автодополнение и др.

Для оптимизации производительности реализованной ИСР были проведены эксперименты с использованием реальных программных проектов с различным размером кодовой базы.

Произведена оценка времени компиляции проекта с использованием стандартного компилятора Go и время компиляции файлов проекта, созданных при помощи разработанной ИСР. Измерение производилось группой экспертов для каждой реализации программного средства. Результаты эксперимента (среднее арифметическое время выполнения компиляции) приведены в табл. 1.

Таблица 1. Сравнение времени компиляции стандартного компилятора Go и модифицированного с разработанным модулем gostr

Компилируемое ПО	Время, с	
	Компилятор Go 1.11.5	Компилятор Go 1.11.5 с модулем gostr
github.com/cloudradar-monitoring/cagent	49,634	40,203 (19 %)
github.com/nikita-vanyasin/go-web-course	30,075	23,760 (23 %)

Произведена оценка времени выполнения рефакторинга – переименование функции [19]. При выполнении данного рефакторинга, среда разработки должна найти все места использования функции (вызовы, ссылки, переопределения) и выполнить переименование. В табл. 2 приведены результаты исследования в ИСР JetBrains GoLand и разработанной ИСР gostr. В результате двадцати проведенных измерений получено среднее арифметическое время выполнения рефакторинга.

Таким образом, осуществленные испытания разработанного программного средства показали:

— уменьшение времени компиляции на 19–23 %;

— уменьшение времени выполнения рефакторинга на 31–34 %.

Таблица 2. Сравнение времени выполнения рефакторинга «переименование функции»

Программный проект	Время, с	
	ИСР «JetBrains GoLand 2019.1»	Разработанная ИСР «gostr»
github.com/cloudradar-monitoring/cagent	1,766	1,154 (34 %)
github.com/nikita-vanyasin/go-web-course	1,650	1,131 (31 %)

Заключение

Проведен анализ состояния работ в области создания интегрированных сред разработки программного обеспечения на основе которого были выделены требования к новой архитектуре интегрированной среды разработки с поддержкой структурного редактирования исходного кода программного обеспечения.

На основе проведенного анализа предложена модифицированная архитектура интегрированной среды разработки со структурным редактором. Разработан и апробирован прототип среды разработки на основе предложенной архитектуры, показавший увеличение эффективности промышленной разработки программного обеспечения за счет увеличения производительности интегрированной среды разработки.

Таким образом, устранены недостатки существующих сред, обозначенные при постановке задачи: отсутствует круговая трансляция из текстового представления исходного кода в промежуточное и обратно; благодаря предложенной архитектуре возможна реализация функциональных возможностей среды разработки, ранее нереализованных в средах разработки со структурным редактированием; увеличена производительность среды разработки с поддержкой структурного редактирования, что увеличивает эффективность промышленной разработки программного обеспечения.

Литература

1. Князева М.А., Тимченко В.А. Структурные редакторы программ на языках программирования высокого уровня и генератор моделей структурных программ в Банке знаний о преобразованиях программ // Искусственный интеллект. 2005. № 4. С. 200–208.
2. Мучник Т.Г. Языково-настраиваемый структурный редактор со средствами семантического контроля // Программирование. 1990. Т. 16. № 2. С. 10–20.
3. Fowler M. Projectional editing. 2008 [Электронный ресурс]. URL: https://martinfowler.com/bliki/ ProjectionalEditing.html (дата обращения: 23.03.2019)
4. Александров С.Ю. Синтаксически-ориентированные редакторы: функциональные возможности и архитектура: Препринт № 3. Новосибирск: ИТМи ВТ АН СССР, 1987. 35 с.
5. Voelter M., Siegmund J., Berger T., Kolb B. Towards user-friendly projectional editors // Lecture Notes in Computer Science. 2014. V. 8706. P. 41–61.
6. Charles Simonyi, Magnus Christerson, Shane Clifford, «Intentional Software», an OOPSLA 2006 paper [Электронный

References

1. Knyazeva M.A., Timchenko V.A. Structural programs editors in the high-level programming languages and generator of structural programs' models within the knowledge bank of programs' reorganization. *Artificial Intelligence Scientific and theoretical journal*, 2005, no. 4, pp. 200–208. (in Russian)
2. Muchnik T.G. A language-adjustable structure editor with means of semantic control. *Programming and Computer Software*. 1990. vol. 16, no. 2, pp. 39–47.
3. Fowler M. *Projectional editing. 2008*. Available at: https://martinfowler.com/bliki/ ProjectionalEditing.html (accessed: 23.03.2019)
4. Aleksandrov S.Yu. Syntax-oriented editors: functionality and architecture: Preprint № 3. Novosibirsk, IPMCE Academy of Sciences of the Soviet Union, 1987, 35 p. (in Russian)
5. Voelter M., Siegmund J., Berger T., Kolb B. Towards user-friendly projectional editors // *Lecture Notes in Computer Science*, 2014, vol. 8706, pp. 41–61.
6. *Charles Simonyi, Magnus Christerson, Shane Clifford, «Intentional Software», an OOPSLA 2006 paper*. Available at: http://www.intentsoft.com/technology/IS_OOPSLA_2006_paper.pdf (accessed: 23.03.2019)

ресурс]. URL: http://www.intentsoft.com/technology/IS_OOPSLA_2006_paper.pdf (дата обращения: 23.03.2019)
7. Reps T., Teitelbaum T. The synthesizer generator // ACM SIGSOFT Software Engineering Notes. 1984. V. 9. N 3. P. 42–48. doi: 10.1145/390010.808247
8. Грачев Д.А., Лаптев В.В. Semantic IDE как обучающая среда и веб-сервис // Математические методы в технике и технологиях-ММТТ. 2013. № 9-1. С. 131–135.
9. Aho A.V., Lam M.S., Sethi R., Ullman J.D. Compilers: principles, techniques, and tools. 2nd edition. Pearson Addison Wesley, 2007. 1009 p.
10. Ванясин Н.В. Семантическое редактирование программного кода в интеллектуальных интегрированных средах разработки приложений // Кибернетика и программирование. 2017. № 1. С. 61–68. doi: 10.7256/2306-4196.2017.1.18881
11. Vaniasin N.V., Sidorkina I.G. Semantic source code editing in Integrated Development Environments // Economics, Management, Information and Technology. 2018. V. 5. N 2. P. 48–53.
12. Clark T. A general architecture for heterogeneous language engineering and projectional editor support // arXiv preprint. 2015. arXiv:1506.03398.
13. Roedy G. Source code in database (Java source code SCID-style browser/editor) [Электронный ресурс]. URL: http://mindprod.com/projects/scid.html (дата обращения: 23.03.2019)
14. Уваров А.Н. Инверсия управления и внедрение зависимостей // Символ науки. 2016. № 10-1. С. 28–32.
15. Немолочнов О.Ф., Зыков А.Г., Поляков В.И., Сидоров А.В. Структурирование программ и вычислительных процессов на множество линейных и условных вершин // Научно-технический вестник Санкт-петербургского государственного университета информационных технологий, механики и оптики. 2005. Т. 5. № 3. С. 207–212.
16. Гришенцев А.Ю., Коробейников А.Г. Улучшение сходимости метода конечных разностей с помощью вычисления промежуточного решения // Научно-технический вестник информационных технологий, механики и оптики. 2012. Т. 12. № 3. С. 124–127.
17. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. AddisonWesley Professional, 1994. 416 p.
18. Hempel B., Lubin J., Lu G., Chugh R. DEUCE: a lightweight user interface for structured editing // Proc. of the 40th International Conference on Software Engineering (ICSE 2018). 2018. P. 654–664. doi: 10.1145/3180155.3180165
19. Мартин Р. Чистый код: создание, анализ и рефакторинг: Пер. с англ. Издательский дом «Питер», 2019. 464 с. (Библиотека программиста)

7. Reps T., Teitelbaum T. The synthesizer generator. *ACM SIGSOFT Software Engineering Notes*, 1984, vol. 9, no 3, pp. 42–48. doi: 10.1145/390010.808247
8. Grachev D.A., Laptev V.V. Semantic IDE as a learning environment and web service. *Mathematical Methods in Technics and Technologies-MMTT*, 2013, no. 9-1, pp. 131–135. (in Russian)
9. Aho A.V., Lam M.S., Sethi R., Ullman J.D. *Compilers: principles, techniques, and tools*. 2nd ed., Pearson Addison Wesley, 2007, 1009 p.
10. Vanyasin N.V. Semantic code editing in intelligent IDEs. *Cybernetics and programming*, 2017, no. 1, pp. 61–68. doi: 10.7256/2306-4196.2017.1.18881 (in Russian)
11. Vaniasin N.V., Sidorkina I.G. Semantic source code editing in Integrated Development Environments. *Economics, Management, Information and Technology*, 2018, vol. 5, no. 2, pp. 48–53.
12. Clark T. A general architecture for heterogeneous language engineering and projectional editor support. *arXiv preprint*, 2015, arXiv:1506.03398.
13. Roedy G. *Source code in database (Java source code SCID-style browser/editor)*. Available at: http://mindprod.com/projects/scid.html (accessed: 23.03.2019)
14. Uvarov A.N. Inversion of control and dependency injection. *Simvol nauki*, 2016, no. 10-1, pp. 28–32. (in Russian)
15. Nemolochnov O.F., Zykov A.G., Poliakov V.I., Sidorov A.V. Structuring of programs and computational processes on the set of linear and conventional vertices. *Scientific and Technical Bulletin of St. Petersburg State University of Information Technologies, Mechanics and Optics*, 2005, vol. 5, no. 3, pp. 207–212. (in Russian)
16. Grishentsev A., Korobeynikov A. Convergence improving of the finite difference method by interim solution calculating. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2012, vol. 12, no. 3, pp. 124–127. (in Russian)
17. Gamma E., Helm R., Johnson R., Vlissides J. *Design Patterns: Elements of Reusable Object-Oriented Software*. AddisonWesley Professional, 1994, 416 p.
18. Hempel B., Lubin J., Lu G., Chugh R. DEUCE: a lightweight user interface for structured editing. *Proc. of the 40th International Conference on Software Engineering (ICSE 2018)*, 2018, pp. 654–664. doi: 10.1145/3180155.3180165
19. Martin R.C. *Clean code: A handbook of agile software craftsmanship*. Prentice Hall, Inc., 2008, 462 p.

Авторы

Ванясин Никита Вадимович — аспирант, Поволжский государственный технологический университет, Йошкар-Ола, 424000, Российская Федерация, ORCID ID: 0000-0001-5968-4361, nikita.vanyasin@gmail.com

Сидоркина Ирина Геннадьевна — доктор технических наук, профессор, декан, Поволжский государственный технологический университет, Йошкар-Ола, 424000, Российская Федерация, ORCID ID: 0000-0002-8414-4946, SidorkinaIG@volgatech.net

Поляков Владимир Иванович — кандидат технических наук, доцент, ординарный доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57190260784, ORCID ID: 0000-0002-2606-5385, v_i_ polyakov@mail.ru

Authors

Nikita V. Vanyasin — Postgraduate, Volga State University of Technology, Yoshkar-Ola, 424000, Russian Federation, ORCID ID: 0000-0001-5968-4361, nikita.vanyasin@gmail.com

Irina G. Sidorkina — D.Sc., Professor, Dean, Volga State University of Technology, Yoshkar-Ola, 424000, Russian Federation, ORCID ID: 0000-0002-8414-4946, SidorkinaIG@volgatech.net

Vladimir I. Polyakov — PhD, Associate Professor, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57190260784, ORCID ID: 0000-0002-2606-5385, v_i_ polyakov@mail.ru