УДК 004.4'23

DOI: 10.15827/0236-235X.129.027-032

Дата подачи статьи: 22.10.19

2020. Т. 33. № 1. С. 027–032

Интегрированная среда разработки с поддержкой структурного редактирования для языка программирования Go

Н.В. Ванясин¹, аспирант, nikita.vanyasin@gmail.com

И.Г. Сидоркина¹, д.т.н., профессор, SidorkinaIG@volgatech.net

¹ Поволжский государственный технологический университет,
г. Йошкар-Ола, 424000, Россия

Одним из главных инструментов для разработчика ПО является интегрированная среда разработки. В статье описывается программная реализация интегрированной среды разработки со структурным редактором, позволяющей создавать и редактировать программные проекты с использованием языка программирования Go. Структурные редакторы дают возможность редактировать исходный код программ как иерархическую структуру вложенных блоков – узлов синтаксического дерева исходного кода, а не набор символов и строк текста.

Программная реализация, рассмотренная в данной статье, отличается от известных способом представления хранимого состояния исходного кода, а также пользовательским интерфейсом структурного редактора, благодаря которому ускоряются действия над исходным кодом. Предложенное решение позволяет повысить производительность среды разработки, а также эффективность работы программиста.

Рассмотрены основные подсистемы и компоненты предлагаемой информационной системы, их назначение и функции. Приведены описание интерфейса взаимодействия с пользователем и его основные отличия.

Одним из основных преимуществ разработанной системы является функция контроля версий, позволяющая отслеживать изменение иерархической структуры кода, а не текста исходного кода, что помогает разработчику ПО получать более детализированную и однозначную информацию при просмотре, слиянии и разрешении конфликтов в истории версий.

В работе представлены результаты экспериментальных исследований производительности интегрированной среды разработки, демонстрирующие эффективность созданной системы. Описаны результаты внедрения программной реализации в производственный и учебный процессы. Сделаны выводы и предположения о возможных перспективах развития структурного программирования для решения промышленных задач.

Ключевые слова: структурное редактирование, среда разработки, интегрированная среда разработки, разработка программного обеспечения, лингвистическое обеспечение, оптимизация производительности.

Известно, что *интегрированная среда разработки* (ИСР) обеспечивает множество функций, таких как компиляция в фоне, автодополнение, навигация по файлам проекта, рефакторинг, отладчик и т.д. Современные ИСР в основном предоставляют текстовые редакторы исходного кода для внесения правок в программный проект, то есть в общем случае разработчик программы работает с символами и строками исходного текста программы. С улучшением программных языков и сред появилась концепция структурных редакторов, когда синтаксическое дерево отображается на экране напрямую [1, 2]. Структурные редакторы позволяют редактировать структуру программы, а не символы и строки текста исход-

ного кода. Также известны семантические редакторы [2], проекционные редакторы [3] и синтаксически-ориентированные редакторы. Структурно любая программа представляется в виде иерархии – *абстрактного синтаксического дерева* (АСД). Это дерево строится на основе исходного текста при помощи сканера и парсера. АСД можно визуально изобразить в виде вложенных блоков. Структурный редактор позволяет пользователю взаимодействовать с АСД напрямую через эти блоки. В этом случае языковые конструкции становятся новыми неделимыми частями программы в отличие от символов и строк текста.

В ходе анализа [4] было выявлено, что существующие реализации сред разработки с

поддержкой структурного редактирования имеют общий недостаток – хранение исходного кода в текстовом представлении, а это приводит к тому, что построение иерархической структуры кода производится каждый раз при загрузке программного проекта, при передаче кода в компилятор или отладчик и т.д. Кроме того, при сохранении кода промежуточное представление транслируется в текстовое, то есть сохраняется в виде текста исходного кода. Такая круговая трансляция сказывается на скорости компиляции и отзывчивости среды разработки – основные параметры, влияющие на эффективность разработки ПО. В ходе исследования было предложено реализовать ИСР с поддержкой структурного редактирования, не обладающую данным недостатком. Создание такой среды дает возможность повысить скорость и эффективность разработки ПО.

Структура ПО

Разработанное программное средство (Свид. о гос. регистр. прог. для ЭВМ № 2019618051) основано на архитектуре, предложенной в [5]. Представленная ИСР позволяет создавать и редактировать ПО на языке программирования Go, поддерживает отладку, рефакторинг, систему контроля версий и другие инструменты для создания ПО.

Разработка выполнена при помощи языков программирования Go, TypeScript и технологии Electron. Выбор языка Go обусловлен множеством факторов: статическая типизация, строгие правила форматирования (задаваемые на уровне компилятора), а также проработанная стандартная библиотека для разбора исходного кода на Go [6].

Программный комплекс включает в себя три основных функциональных блока:

- внутренний функциональный блок среды разработки – backend, реализующий всю логику работы с исходным кодом ПО, интер-

фейс для компилятора и отладчика, систему контроля версий и другие функции ИСР;

- модуль для компилятора и отладчика языка Go, который позволяет компилятору использовать промежуточное представление (CodeDOM) [7] исходного кода в качестве входных данных;

- пользовательский интерфейс для взаимодействия пользователя со средой разработки – frontend, предоставляющий 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.

На рисунке 1 показаны структура программной системы и взаимодействие ее с пользователем и компилятором Go.

Пользовательский интерфейс разработан в соответствии с принципами, предложенными в [8]. Внешний вид пользовательского интерфейса представлен на рисунке 2.

На рисунке отображены основные компоненты пользовательского интерфейса.

- В верхней части расположено главное меню приложения. Оно позволяет пользователю запустить все возможные команды в рамках текущего контекста и состоит из набора подменю, сгруппированных по логике действий: Проект, Редактирование, Поиск, Вид, Код, Рефакторинг, Запуск, Помощь.

- В левой части расположен обзор структуры проекта (Solution Explorer). Он позволяет пользователю видеть и редактировать структуру проекта. Возможна организация модулей проекта в виде вложенных папок. При помощи двойного клика можно открыть модуль для редактирования.

- В центральной части размещена основная рабочая область редактора – список вкладок с исходным кодом ПО. В состав каждой открытой вкладки с кодом входят:

Рис. 1. Диаграмма взаимодействия

Fig. 1. An interaction diagram

Рис. 2. Экранная форма пользовательского интерфейса разработанной ИСР со структурным редактором

Fig. 2. User interface screenshot of the created IDE with structured editing support

- компонент «хлебные крошки» для отображения текущего местоположения курсора в иерархической структуре модуля (доступна навигация при помощи клика по элементам компонента);
- таблица импорта, отображающая список зависимых пакетов для текущего модуля (в основном формируется автоматически, но пользователь может задать альтернативное имя для импорта (алиас), а также создать «безымянный» импорт);
- область редактирования, содержащая сам структурный редактор для АСД модуля (навигацию и действия над грамматическими конструкциями можно выполнять как с клавиатуры, так и мышью).
- В нижней части расположена панель для управления отображением дочерних окон, таких как «Запуск», «Отладка», «Система контроля версий» и «Результаты поиска».

Обоснование эффективности ИСР с использованием структурного редактирования исходного кода

Для определения производительности реализованной ИСР были проведены экспери-

менты с использованием реальных данных (исходного кода проектов ПО).

Оценивались время компиляции проекта с использованием стандартного компилятора Go и время компиляции файлов проекта, созданных при помощи разработанной ИСР. Измерение производилось группой экспертов [9] для каждой реализации программного продукта. Результаты эксперимента (среднее арифметическое время выполнения компиляции) приведены в таблице 1.

Произведена оценка времени выполнения рефакторинга «переименование функции» [10]. При выполнении данного рефакторинга среда разработки должна найти все места использования функции (вызовы, ссылки, переопределения) и осуществить переименование. В таблице 2 приведены результаты исследования в ИСР JetBrains GoLand и разработанной ИСР. В результате нескольких десятков измерений получено среднее арифметическое времени выполнения рефакторинга.

Для определения эффективности реализованного в ИСР метода контроля версий был проведен эксперимент с участием 25 начинающих и профессиональных разработчиков ПО. Создана экспериментальная установка – персо-

Таблица 1

Сравнение времени компиляции стандартным компилятором Go и модифицированным

Table 1

Comparison of compilation time by Go compiler and modified Go compiler

Компилируемое ПО	Компилятор Go 1.11.5(время, сек.)		Эффектив-ность
	Стандартный	С разработанным модулем	
github.com/cloudradar-monitoring/cagent	49,634	40,203	19 %
github.com/nikita-vanyasin/go-web-course	30,075	23,760	23 %

Таблица 2

Сравнение времени выполнения рефакторинга «переименование функции»

Table 2

Comparison of “function rename” execution time

Программный проект	ИСП JetBrains GoLand 2019.1 (время, сек.)	Разработанная ИСП (время, сек.)	Эффектив-ность
github.com/cloudradar-monitoring/cagent	1,766	1,154	34 %
github.com/nikita-vanyasin/go-web-course	1,650	1,131	31 %

Таблица 3

Результаты определения семантики изменений при помощи системы контроля версий

Table 3

Results of identifying the semantics of changes using version control system

Программный проект	ИСП JetBrains GoLand 2019.1		Разработанная ИСП	
	Среднее время решения	Правильные ответы	Среднее время решения	Правильные ответы
github.com/cloudradar-monitoring/cagent	14 мин. 14 сек.	85 %	5 мин. 40 сек.	98 %
github.com/nikita-vanyasin/go-web-course	12 мин. 45 сек.	93 %	4 мин. 20 сек.	99 %

нальный компьютер с установленными ИСП JetBrains Goland и разработанной ИСП. В ходе эксперимента участник должен был попытаться определить семантику изменений в исходном коде проекта при помощи установленной ИСП. Время эксперимента было ограничено 30 минутами, участник должен был ответить на 6 вопросов теста об истории изменений. Результаты эксперимента представлены в таблице 3.

Испытания разработанного программного средства показали:

- уменьшение времени компиляции на 19–23 %;
- уменьшение времени выполнения рефакторинга на 31–34 %;
- более высокую детализацию истории версий исходного кода ПО, что позволяет уско-

пить определение семантики изменений программистом на 75 %, а также повысить корректность на 6–13 %.

Полученные результаты демонстрируют эффективность предложенного решения.

Разработанная ИСП внедрена в производственный процесс компаний iSpring и Travel-Line, а также в учебный процесс Поволжского государственного технического университета.

Заключение

В ходе работы была реализована ИСП с поддержкой структурного редактирования для языка Go. Разработанное программное средство основано на архитектуре, предложенной в [5]. Проведены серии экспериментов, под-

твердивших улучшение значений параметров, влияющих на скорость и эффективность разработки промышленного ПО в созданной ИСР с поддержкой структурного редактирования исходного кода.

На данном этапе реализуется поддержка других языков программирования, а также визуального программирования для промышленного применения, что позволит расширить область использования разработанной ИСР.

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. № 2. С. 10–20.
3. Fowler M. ProjectionalEditing. 2008. URL: <https://martinfowler.com/bliki/ProjectionalEditing.html> (дата обращения: 20.10.2019).
4. Ваясин Н.В., Сидоркина И.Г. Анализ функциональных возможностей структурных редакторов исходного кода программного обеспечения // Вестн. ПГТУ: Радиотехнические и инфокоммуникационные системы. 2019. № 1. С. 64–77. DOI: 10.25686/2306-2819.2019.1.64.
5. Vanyasin N.V., Sidorkina I.G. Semantic source code editing in integrated development environments. EMIT, 2018, vol. 5, no. 2, pp. 48–53.
6. Donovan A.A., Kernighan B.W. The Go Programming Language. Addison-Wesley, 2015, 380 p.
7. Aho A.V., Lam M.S., Sethi R., Ullman J.D. Compilers: principles, techniques & tools. 2007, Addison Wesley, 1000 p.
8. Voelter M., Siegmund J., Berger T., Kolb B. Towards user-friendly projectional editors. Proc. Intern. Conf. on Software Language Engineering. Springer, Cham, 2014, pp. 41–61. DOI: 10.1007/978-3-319-11245-9_3.
9. Орлов А.И. Экспертные оценки. М., 2002. 486 с.
10. Мартин Р. Чистый код: создание, анализ и рефакторинг. Библиотека программиста. СПб, 2013. 464 с.

Software & Systems
DOI: 10.15827/0236-235X.129.027-032

Received 22.10.19
2020, vol. 33, no. 1, pp. 027–032

Integrated development environment with structured editing support for Go programming language

*N.V. Vanyasin*¹, *Postgraduate Student, nikita.vanyasin@gmail.com*

*I.G. Sidorkina*¹, *Dr.Sc. (Engineering), Professor, SidorkinaIG@volgatech.net*

¹ *Volga State University of Technology, Yoshkar-Ola, 424000, Russian Federation*

Abstract. The paper describes software implementation of an integrated development environment with structured editing support that allows creating and editing software projects using Go programming language. Structure editors allow editing a source code as a hierarchy of embedded blocks in opposite to text editors, which allow editing a source as a list of characters and lines.

The proposed software implementation is different from the known systems by the storage representation method that allows speeding up the performing operations with a source code. The proposed editor also allows increasing the overall efficiency of a software development process. This implementation solves the problem of circular translation of a source code into an intermediate representation and vice versa, which is common among modern structured editors of a source code.

The solution for this problem improves the development environment performance and programmer's productivity in several ways. The study lists the basic subsystems and components of the proposed information system, their purpose, functions and work principle.

The paper describes the user interface and its basic features. One of the main features of the proposed implementation is a version control system that allows tracking the differences between source code hierarchical structures, not the differences between text files. It allows a software developer to get more detailed and unambiguous information when browsing a version history as well when performing merge or conflict resolution.

The paper presents test results of the integrated development environment performance and the effectiveness of software development using this environment. It also describes the implementation results of the proposed system in the development process of industrial companies and educational organization. Finally, there are some insights on further development of the study.

Keywords: structured editing, development environment, integrated development environments, software development, linguistic support system, performance optimization.

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' re-organization. *Artificial Intelligence*. 2005, no. 4, pp. 200–208 (in Russ.).
2. Muchnik T.G. A language-adjustable structure editor with means of semantic control. *Programming and Computer Software*. 1990, no. 2, pp. 10–20 (in Russ.).
3. Fowler M. *Projectional Editing*. 2008. Available at: <https://martinfowler.com/bliki/ProjectionalEditing.html> (accessed October 20, 2019).
4. Vanyasin N.V., Sidorkina I.G. Analysis of functionality of structure source code editors. *Sci. Tech. Volga Region Bull.* 2019, no. 1, pp. 64–77 (in Russ.). DOI: 10.25686/2306-2819.2019.1.64.
5. Vanyasin N.V., Sidorkina I.G. Semantic source code editing in integrated development environments. *EMIT*. 2018. vol. 5, no. 2, pp. 48–53.
6. Donovan A.A., Kernighan B.W. *The Go Programming Language*. Addison-Wesley Professional, 2015, 380 p.
7. Aho A.V., Lam M.S., Sethi R., Ullman J.D. *Compilers: Principles, Techniques & Tools*. 2007, 1000 p.
8. Voelter M., Siegmund J., Berger T., Kolb B. Towards user-friendly projectional editors. *Proc. Intern. Conf. Software Language Engineering*. Springer, Cham, 2014, pp. 41–61. DOI: 10.1007/978-3-319-11245-9_3.
9. Orlov A.I. *Expert Evaluation*. Moscow, 2002, 486 p. (in Russ.).
10. Martin R. *Clean code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, Inc., 2008, 462 p. (Russ. ed.: St. Petersburg, 2013, 464 p.).

Для цитирования

Ванясин Н.В., Сидоркина И.Г. Интегрированная среда разработки с поддержкой структурного редактирования для языка программирования Go // Программные продукты и системы. 2020. Т. 33. № 1. С. 027–032. DOI: 10.15827/0236-235X.129.027-032.

For citation

Vanyasin N.V., Sidorkina I.G. Integrated development environment with structured editing support for Go programming language. *Software & Systems*. 2020, vol. 33, no. 1, pp. 027–032 (in Russ.). DOI: 10.15827/0236-235X.129.027-032.