VERIFICATION OF DOCUMENTS AND INFORMATION USING INFORMATION SYSTEM OF CUSTOMS AUTHORITIES

I.M. Tarasova, I.M. Kolodkin

Russian customs academy Vladivostok branch,
Vladivostok, Russia
Far Eastern Federal University,
Vladivostok, Russia

The article interaction of participants in foreign economic activity and customs officials is analyzed, and a model for verifying documents and information using the customs information system based on UML diagrams is visualized. The result of the work is the generated UML diagrams of the verification process of documents and information using the information system of customs authorities.

Keywords: UML modeling, document verification, information verification, information system.

УДК 004.4'24

ПРИМЕНЕНИЕ МЕТОДОВ ВАЙБ-КОДИНГА ДЛЯ РЕФАКТОРИНГА СУЩЕСТВУЮЩИХ ПРОЕКТОВ

А.Ю. Тимонин

Пензенский государственный университет,
г. Пенза, Россия

В статье рассматриваются перспективы использования современных нейронных сетей, основывающихся на больших языковых моделях, для автоматизации написания актуализированного программного кода в рамках разработки новых версий различного программного обеспечения. Приводятся преимущества и недостатки методов генерации программного кода по сравнению с традиционными средствами написания кода, когда программист импровизирует, пользуясь своим опытом, либо использует чужие наработки, готовые шаблоны и алгоритмы.

Ключевые слова: вайб-кодинг, рефакторинг кода, искусственный интеллект, большие языковые модели, автоматизация программирования.

Развитие генеративных нейронных сетей стало трендом текущего десятилетия. Существующие проекты искусственного интеллекта (ИИ) уже позволяют

генерировать логично построенные тексты, аудиовизуальные образы и их комбинации на заданные темы [1]. Среди перспективных областей применения ИИ выделяется автоматизация разработки программного обеспечения (ПО), в том числе для рефакторинга (актуализации, переписывания и оптимизации кода) уже существующих программных продуктов [2]. Внедрением таких методов разработки занимаются как профессионально в крупных компаниях (МТС, Сбер, Яндекс и т. п.), так и в частном порядке различными разработчиками, в том числе учащимися вузов в ходе выполнения практических, лабораторных, курсовых и дипломных работ.

Термин «вайб-кодинг» (англ. vibe coding) обозначает метод программирования, использующий большие языковые модели и искусственный интеллект для генерации программного кода на основе описания задачи на естественном языке, и впервые официально был озвучен в феврале 2025 года учёным в области машинного обучения Андреем Карпатым [3]. Слово «вайб» в переводе с английского обозначает определенную атмосферу, настроение, которое исходит от чего или кого-либо. Сам метод начал использоваться с тех пор, когда GPT-подобные нейронные сети стали обучаться на множестве открытых источников с исходными кодами компьютерных программ.

По своей сути это автоматическая генерация программного кода, основанная не на опыте разработчика или композиции/декомпозиции готовых алгоритмов стандартных задач, а на попытке нейронной сети сопоставить словесное описание поставленной задачи с имеющимися у неё готовыми примерами решений подобных задач. Методы вайб-кодинга являются эволюционным развитием подходов программистов к автоматизации разработки ПО и упрощения человеко-машинного взаимодействия.

Первой вехой в автоматизации информационных задач можно считать появление программируемых вычислительных машин и первых алгоритмических языков. В дальнейшем языки программирования эволюционировали, произошло разделение на низкий и высокий уровни:

– более понятные для компьютера и имеющие больше возможностей по оптимизации кода и эффективного использования аппаратного обеспечения ЭВМ;

– более понятные для разработчика, что позволило ускорить разработку программ, появились возможности для переноса ПО на различные аппаратные платформы и т.д.

Затем стали развиваться инструменты для написания кода. В специализированных средах IDE появились поддержка кросс-компиляции, комбинирования кода на различных языках, возможности автоматической проверки кода на ошибки и его оптимизации в ходе компиляции/трансляции. Расширились возможности использования расширяемых наборов библиотек и модулей языков программирования. Особо выделился язык Python с его обширной библиотекой расширений, что обусловило его популярность в различных областях, начиная с обучения программированию и заканчивая передовыми разработками в Data Science и машинном обучении.

Следующим важным шагом в автоматизации разработки ПО стали средства визуального программирования, изначально создававшиеся для обучения азам программирования (язык ЛОГО как пример), но в дальнейшем оказавшиеся частью сред разработки для создания графических интерфейсов, UML-диаграмм и схем данных.

Помимо технических предпосылок для появления вайб-кодинга имеются и социально-коммуникативные предпосылки. Изначально сообщества разработчиков были по большой части закрытыми, ограниченными организациями, в которых работали разработчики, и академической средой. С развитием Интернета стало больше возможностей для обмена идеями алгоритмов и их реализаций на различных языках программирования. Начали развиваться сообщества разработчиков, занятые коллективной разработкой открытого ПО. Помимо информационно-справочных ресурсов, посвященных программированию, появились различные сайты сообществ программистов (такие как cyberforum.ru, github.com, stackoverflow.com). Изначально их аудитория состояла из профессионалов, но с повышением доступности Интернета, основной аудиторией таких сайтов стали студенты технических вузов и начинающие разработчики. С середины 10-ых годов ввиду популярности таких сайтов и наполняемости их кодовой базы, копирование кода стало использоваться повсеместно. Свой вклад внесло и развитие концепции свободного и открытого ПО, когда исходный код публикуется в открытом доступе, а его разработка ведется сообществом разработчиков коллективно.

В ходе обучения новейших генеративных сетей на базе вышеперечисленных ресурсов весь имеющийся опыт разработчиков оказался сохранен внутри моделей искусственного интеллекта, что позволяет обращаться к нему любому пользователю данных моделей без какой-либо первоначальной технической подготовки, что и стало причиной популярности вайб-кодинга.

Всё большая необходимость в переписывании (рефакторинге) существующего программного кода с целью его актуализации обусловлен рядом причин. Происходит постоянное обновление различных программных библиотек, модулей и зависимостей, выявление уязвимостей и необходимость внедрения нового функционала в существующие программные продукты. Получают популярность новые языки программирования, такие как Rust, Go, Mojo, отличающиеся от классических улучшенной работой с динамически выделяемой памятью, параллелизмом и сборкой мусора, тестируемостью, поддержкой сторонних пользовательских библиотек. Это требует от разработчиков дополнительных затрат ресурсов на чтение готового кода, выявления в нём проблемных частей, особенно если код был написан давно, либо на другом языке или старой версии того же языка, либо изначальный автор кода уже не занимается его поддержкой. Также бывают ситуации, когда исходные коды повреждены или утрачены.

Необходимо уменьшить затраты на переписывание, в идеале, чтобы это происходило автоматически. В качестве примера большого проекта можно привести ОС Windows, содержащую огромное количество legacy-кода из прошлых версий, начиная с 90-ых годов. Актуализация кода является большой проблемой

даже для такой крупной корпорации как Microsoft, т.к. обновление компонентов системы занимает большое количество времени и часто приводит к появлению трудноотслеживаемых непредвиденных ошибок. Другим примером является ядро Linux, объем кода которого растет в геометрической прогрессии и уже превысил возможности отдельного человека по его изучению. Несмотря на строгие регламенты по принятию изменений процесс переписывания отдельных модулей с языка C на Rust происходит медленно и вызывает споры в сообществе ввиду несоразмерности трудозатрат и теоретической выгоды от использования преимуществ нового языка. Разработка кроссплатформенного ПО и поддержка множества устройств на архитектуре ARM [4] также показывает необходимость в автоматизации разработки ПО, в чем может помочь вайб-кодинг.

К плюсам вайб-кодинга можно отнести уменьшение трудозатрат на выполнение рутинной сложной и повторяющейся работы, что позволяет разработчикам сконцентрироваться на самой задаче, а не особенностях ее технической реализации. Также это позволяет быстро подготовить несколько вариантов решения задачи, выбрать наилучший и уже работать только с ним. Еще одним преимуществом вайб-кодинга является возможность профессиональному программисту ускоренно начать работу с незнакомым ему языком программирования.

Однако помимо опытных разработчиков вайб-кодинг используют и начинающие. Это несёт в себе следующие угрозы:

– ввиду отсутствия практического опыта в разработке комплексного ПО они не в состоянии разобраться и вручную довести до ума сгенерированный код;

– у них не развиваются навыки алгоритмизации и самостоятельного написания кода, работы в коллективе.

Ввиду этого могут быть пропущены существенные ошибки, код может оказаться неоптимальным и содержать скрытые уязвимости. Зачастую нейронные сети галлюцинируют, и могут генерировать правдоподобно выглядящий, но бессмысленный или некорректный код, который бывает трудно распознать даже опытному программисту. Стоит обратить внимание, что некоторые языковые модели обучаются не только на открытых проектах, которые в целом могут иметь в себе уязвимости и бэкдоры, но и проприетарных проектах, защищённые коммерческой тайной. Это создает правовые угрозы для разработчиков. В частности, можно привести пример ReactOS, разработчики которой обязаны быть не ознакомлены с исходными кодами ОС Windows. Кроме того, законодательство некоторых государств запрещает проводить дизассемблирование стороннего программного кода в том числе и в ознакомительных целях.

Можно заключить, что вайб-кодинг является новым этапом в автоматизации разработки ПО. В перспективе, деятельность программистов несколько видоизменится: вместо разработки детализированных алгоритмов с нуля и написания кода, учитывающего специфику языка программирования, они будут автоматизировано собирать в целостную систему компоненты кода из генерируемых шаблонов, выбирая из множества предложений наиболее оптимальные для их задач. Для этого разработчикам нужно будет уметь работать с чужим кодом, ре-

шать конфликты зависимостей в наименовании переменных, выборе типов данных, количестве передаваемых аргументов, проверять исходный код на адекватность. Методы вайб-кодинга могут уменьшить затраты на «черновую» работу, обычно поручаемую стажерам и джуниор-разработчикам, создание минимально допустимо функционирующего кода, тестовых проектов и несложных программных модулей полностью перейдет к искусственному интеллекту. Это повысит минимальные требования к профессиональным компетенциям и навыкам разработчиков, т. е. по сути останется полностью актуальной деятельность миддлов и сеньоров.

Список использованных источников

1. Ключевые возможности генеративных систем для создания образовательных материалов // Сбер Университет. URL: https://courses.sberuniversity.ru/ai-education/5/1

2. Куликов А.А., Шуралев С.И. Проектирование нейронной сети для рефакторинга кода // ББК 1 Н 34. – С. 5735.

3. Edwards B. Will the future of software development run on vibes? // Ars Technica URL: https://arstechnica.com/ai/2025/03/is-vibe-coding-with-ai-gnarly-or-reckless-maybe-some-of-both/

4. Тимонин А.Ю., Макеев Д.И., Филиппов О.Р. Трудности разработки свободного системного программного обеспечения для устройств на архитектуре ARM // Новые информационные технологии и системы (НИТиС-2023) : Сборник научных статей по материалам XX Международной научно-технической конференции, посвященной 80-летнему юбилею Пензенского государственного университета, Пенза, 16–17 ноября 2023 года. – Пенза: Пензенский государственный университет, 2023. – С. 191-195. – EDN BXSVMC.

USING VIBE CODING TECHNIQUES TO REFACTOR EXISTING PROJECTS

A.Yu. Timonin

Penza State University,
Penza, Russia

The article discusses the prospects for the use of modern LLM neural networks to automate the writing code as part of the iterative software development. The advantages and disadvantages of generating software methods compared to traditional programming means, when the programmer improvises, uses own experience, or other ones achievements, or ready-made templates and algorithms.

Keywords: vibe coding, code refractoring, artificial intelligence, large language models, programming automation.