МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет компьютерных наук

Кафедра информационных технологий управления

Эссе

Системная инженерия на примере системы «Visual Studio Code»

09.04.02 Информационные системы и технологии

Информационные технологии в менеджменте

Руководитель _____*С.Д. Махортов, д.т.н., профессор* __. __.20__

Обучающийся _____*М.Н. Котолевский, 1 курс, д/о*

Воронеж 2023

Программирование – это процесс создания программного обеспечения, которое позволяет компьютеру выполнять определенные задачи. Оно является неотъемлемой частью современной технологической индустрии и широко используется в различных сферах, таких как разработка веб-сайтов, мобильных приложений, игр, искусственный интеллект, робототехника. С каждым днем количество задач, решаемых с помощью программирования, увеличивается, и появляется все больше возможностей для улучшения процесса разработки программного обеспечения.

Одним из самых важных инструментов для программистов является редактор кода, который позволяет создавать, редактировать и отлаживать код. Сегодня на рынке представлено множество редакторов кода, но одним из наиболее популярных инструментов для разработки программного обеспечения является Visual Studio Code (в дальнейшем будем использовать сокращение VS Code), разработанный компанией Microsoft. Этот программный продукт представляет собой сложную систему, а значит, может быть рассмотрен с точки зрения системной инженерии.

Для начала определим цель системы (функция) и элементы, из которых система состоит (конструкция). Сама система представляет собой единство данных понятий.

Основные функции системы:

1. **Редактирование и создание кода:**
    - Подсветка синтаксиса и автодополнение кода.
    - Отладка кода и пошаговое выполнение.
    - Рефакторинг кода и оптимизация импортов.
    - Интеграция с Git и другими системами контроля версий.
2. **Расширяемость и интеграции:**
    - Установка расширений для работы с языками и технологиями.
    - Интеграция со сборщиками проектов и инструментами CI/CD.
3. **Управление проектами и файлами:**
    - Просмотр структуры проекта и файлов.

- Интеграция с локальными и удаленными файловыми системами.
- Поиск и замена текста в файлах и проектах.

4. **Работа в команде:**
   - Совместная работа над проектами в реальном времени.
   - Инструменты коллаборации, например, LiveShare.
   - Обмен кодом и файлами через каналы связи, такие как Slack или Microsoft Teams.
5. **Инструменты для улучшения производительности:**
   - Настройка параметров редактора.
   - Использование горячих клавиш для ускорения работы.
   - Работа в нескольких окнах одновременно и возможность разделения экрана на несколько областей.
6. **Поддержка различных языков и платформ:**
   - Поддержка большого количества языков программирования, включая Python, JavaScript, C#, Java.
   - Поддержка работы на различных операционных системах, включая Windows, macOS и Linux.

Конструкцией системы являются:

1. **Каркас редактора**, который предоставляет интерфейс пользователя, позволяющий открыть, редактировать и сохранять файлы.
2. **Ядро редактора**, которое обеспечивает основную функциональность, такую как подсветка синтаксиса, автодополнение кода, поиск и замена текста, управление макросами.
3. **Модуль расширений**, который позволяет разработчикам добавлять новые функции и возможности в редактор через написание плагинов.

4. **Модуль управления проектами и файлами**, который обеспечивает возможность просмотра структуры проекта и файлов, интеграцию с различными файловыми системами и возможность поиска и замены текста.
5. **Модуль интеграции с системами контроля версий**, который обеспечивает поддержку различных систем контроля версий, таких как Git, и позволяет выполнить все основные операции, такие как коммит, откат, переключение веток и многое другое.

Теперь определим архитектуру системы. VS Code состоит из многоуровневого и модульного ядра, которое можно нарастить с помощью расширений, как показано на рисунке 1. Расширения запускаются в отдельном процессе, называемом Extension Host, который использует Extension API.

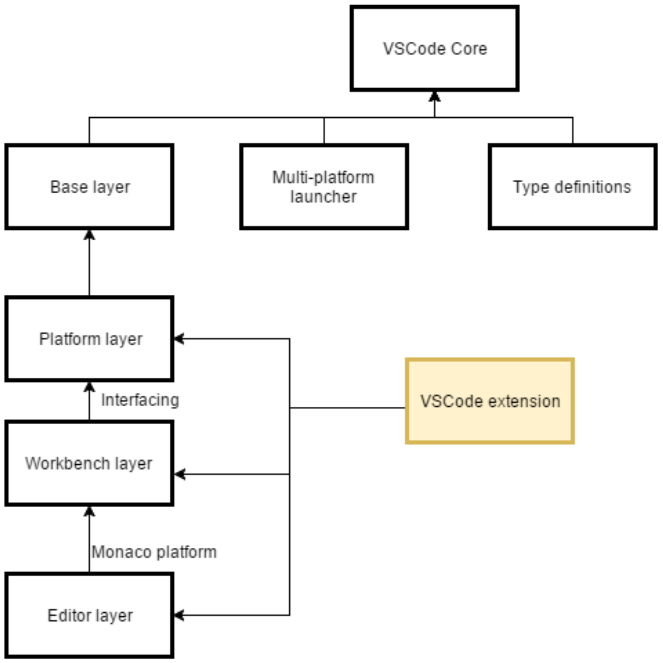

Рисунок 1.

Ядро Visual Studio Code разделено на следующие уровни:

- *base:* Предоставляет общие утилиты и стандартные блоки пользовательского интерфейса.

- *platform:* определяет поддержку внедрения сервисов и базовые сервисы для Visual Studio Code.
- *editor*: редактор Монако доступен как отдельный загружаемый компонент.
- *workbench*: принимает редактор Монако и предоставляет основу для вьюлетов, таких как проводник, строка состояния или строка меню, используя Electron для реализации настольного приложения Visual Studio Code.

VS Code включает в себя основной процесс и процесс рендеринга. В то же время, поскольку VS Code предоставляет возможность расширения подключаемых модулей, а также в целях безопасности и стабильности, на рисунке 2 показан еще один Extension Host. Фактически, этот узел расширения также является независимым процессом для запуска нашего кода подключаемого модуля. И, как и процесс рендеринга, они независимы друг от друга. Процесс хостинга расширения предоставляет разработчикам подключаемых модулей API-интерфейсы VS Code для использования.

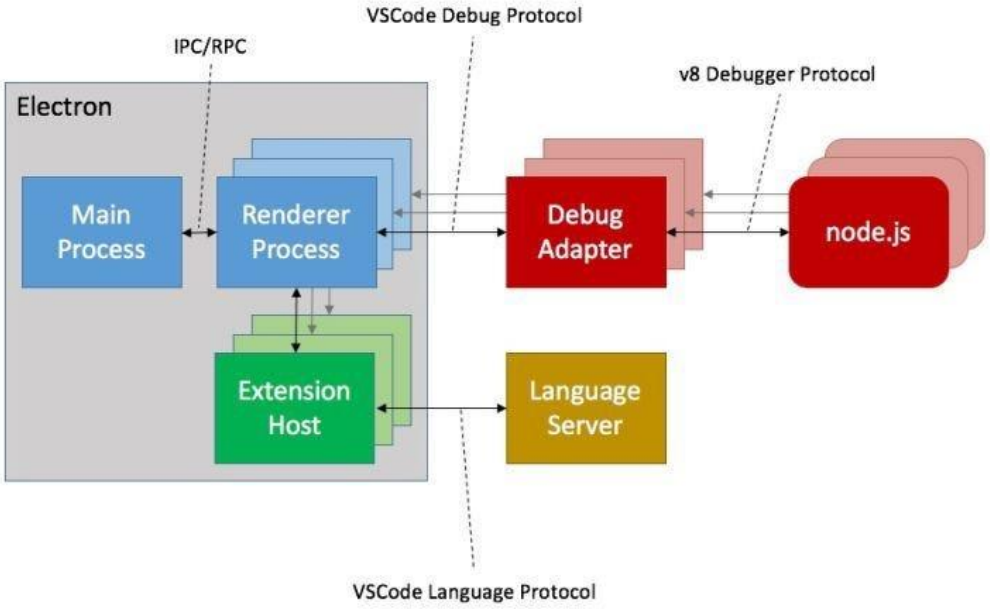

Рисунок 2.

Определен набор правил, которым должна подчиняться каждая часть:

- Не может быть никакой зависимости извне *vs/workbench/parts* в *vs/workbench/parts.*

- Каждая часть должна предоставлять свой внутренний API из одного файла (например, *vs/workbench/parts/search/common/search.ts*).
- Часть может зависеть от внутреннего API другой части (например, часть git может зависеть от *vs/workbench/parts/search/common/search.ts*).
- Часть никогда не должна проникать внутрь другой части (внутреннее — это все то, что находится внутри части, чего нет в одном общем файле API).

Для данной системы можно выделить несколько стейкхолдеров-заинтересованных лиц. Предпочтения, цели и стремления заинтересованных лиц определяют требования к системе. К стейкхолдерам системы VS Code можно отнести:

- **Пользователи.** Люди, которые используют VS Code для написания кода и разработки приложений. Они заинтересованы в том, чтобы VS Code была простой в использовании, быстрой, надежной, масштабируемой и имела широкий спектр функций, чтобы помочь им в их работе.
- **Разработчики.** Люди, которые создают и поддерживают VS Code. Они заинтересованы в том, чтобы система была легко расширяема, имела открытый исходный код, поддерживала плагины и различные интеграции с другими системами.
- **Компании.** Компании, которые используют VS Code в качестве средства разработки, заинтересованы в том, чтобы система была эффективной и помогала им быстро создавать качественное программное обеспечение.
- **Партнеры.** Партнеры, которые создают интеграции и расширения для VS Code, заинтересованы в том, чтобы система была легко расширяема и имела большую аудиторию пользователей.

- **Образовательные учреждения.** Учреждения, которые используют VS Code для обучения студентов программированию, заинтересованы в том, чтобы система была доступна, проста в использовании и имела обширную документацию.

Отметим ограничения, которые накладываются на данную систему:

- **Производительность**. VS Code не должна быть медленной при работе с очень большими проектами, особенно если у пользователя ограниченные ресурсы компьютера.
- **Ограниченные возможности визуального интерфейса**. VS Code разработан в первую очередь для работы с кодом, что может означать, что у него не так много функций визуального интерфейса, как у других сред разработки.
- **Ограниченная поддержка языков.** VS Code поддерживает большинство популярных языков программирования, но его поддержка для менее распространенных языков может быть ограничена или неполной.
- **Хранение и обработка больших объемов данных.** VS Code может ограничивать возможности хранения и обработки больших объемов данных, таких как огромные проекты с множеством файлов.
- **Операционная система.** VS Code имеет поддержку для большинства популярных операционных систем, таких как Windows, Linux и MacOS.
- **Безопасность.** VS Code, как и любое другое программное обеспечение, может подвергаться угрозам безопасности, таким как вредоносный код, уязвимости в коде.

VS Code можно рассматривать как часть компании Microsoft и как один из ее продуктов. В таком случае, можно представить систему VS Code в виде **холархии** следующим образом:

1. Уровень **Microsoft Company** (верхний уровень) - представляет собой компанию Microsoft, которая разрабатывает и выпускает различные программные продукты.
2. Уровень **Microsoft Products** - включает в себя различные продукты компании Microsoft, такие как Windows, Office, Azure и т.д.
3. Уровень **VS Code** - представляет собой среду разработки, разработанную компанией Microsoft.
4. Уровень **компоненты VS Code** - включает в себя различные компоненты, необходимые для функционирования VS Code, такие как редактор кода, система контроля версий.

Каждый уровень системы зависит от уровня ниже и обеспечивает интерфейс для уровня выше. Например, компоненты VS Code зависят от самой среды разработки, а сама среда разработки зависит от продуктов Microsoft. В целом, система VS Code является одним из продуктов компании Microsoft и обеспечивает пользователям удобную среду для разработки программного обеспечения.

Теперь рассмотрим обеспечивающие системы и системы в эксплуатационной среде.

- **Обеспечивающие системы**: компания Microsoft, разработчики Microsoft Products, сторонние разработчики, программный и аппаратный инструментарий разработчиков.
- **Системы в эксплуатационной среде**: организации, конечные пользователи, компьютер.

Перейдем к рассмотрению жизненного цикла (последовательности стадий развития) данной системы:

1. **Замысел.** Microsoft предлагает создание новой среды разработки под названием Visual Studio Code, которая предназначается для обеспечения комфортной и продуктивной работы программистов. Исследование рынка и конкурентных решений показывает, что существующие

решения имеют ограничения, и требуется новый подход к созданию IDE. Определение функциональных и нефункциональных требований к системе, которые включают в себя поддержку различных языков программирования, возможность интеграции с другими инструментами, высокую производительность и открытый исходный код.

2. **Разработка.** Создается архитектура системы, которая обеспечивает высокую производительность и гибкость в работе с различными языками программирования. Разрабатывается дизайн пользовательского интерфейса, который обеспечивает удобство и продуктивность в работе с системой. Разрабатываются алгоритмы и структуры данных, обеспечивающие быстрый и эффективный поиск и редактирование кода. Описанная архитектором система документируется и передается разработчикам для дальнейшей реализации.
3. **Производство.** Система реализуется на основе полученной документации. VS Code является программным продуктом и выпускается в виде бесплатного программного обеспечения. Производство включает в себя сборку и распространение программного обеспечения через официальный сайт.
4. **Использование.** VS Code используется программистами для разработки и отладки программного кода. Она поддерживает различные языки программирования, фреймворки и инструменты.
5. **Поддержка.** Microsoft выпускает регулярные обновления VS Code, которые исправляют ошибки и улучшают производительность системы. Существует официальная документация и сообщество разработчиков, которые предоставляют поддержку и помощь в работе с системой.
6. **Прекращение использования.** На данной стадии VS Code теряет конкурентоспособность, имеет проблемы с производительностью или стабильностью, новые технологии заменяют VS Code в качестве

предпочтительной среды разработки или проблемы с лицензированием так же могут привести к прекращению использования.

Согласно **ISO 15228:2008** существует 4 типа практик: контрактации, проектные, обеспечение проектов и технические. Рассмотрит список технических практик системной инженерии:

1. **Сбор требований:** Данная практика подразумевает определение заинтересованных лиц и оформления списка требования путем опроса пользователей. Производится анализ рынка, существующих решений, а также анализ средств реализации. Например, пользователи могут запрашивать поддержку новых языков программирования, возможность отладки кода на удаленных серверах и т.д.
2. **Анализ требований:** После сбора требований идет их анализ, чтобы определить, какие из них могут быть реализованы, какие недостаточно точны и требуют дополнительной спецификации и какие требования не могут быть выполнены. Например, если пользователи запросили поддержку нового языка программирования, команда разработчиков проводит исследования, чтобы узнать, насколько это возможно и насколько сложно это будет реализовать.
3. **Архитектурный дизайн**: Данная практика подразумевает проектирование архитектуры системы, определение структуры, компонентов и взаимодействие между ними. Например, если требуется добавить поддержку контроля версий, команда разработчиков проектирует компоненты, необходимые для ее реализации, и определяет, как эти компоненты будут взаимодействовать между собой.
4. **Изготовление:** Данная практика подразумевает создание конкретных модулей системы. Например, модуль управления проектами и файлами.
5. **Интеграция**: После того, как компоненты системы были созданы, их необходимо интегрировать в единую систему. Это может включать сборку, конфигурирование и настройку системы. Например, объединяются модуль управления проектами и файлами и модуль

интеграции с системами контроля версий, который вдальнейшем объединяется с другими модулями системы и получается единая система VS Code.

6. **Верификация:** Данная практика подразумевает проверку системы установленным требованиям. Например, корректность работы на ОС Windows и Linux различных версий, корректность импорта и экспорта файлов.
7. **Переход к эксплуатации:** Данная практика подразумевает выполнение определенных этапов перед ее использованием конечными пользователями. Например, загрузка актуальной версии на основной сайт.
8. **Валидация:** Данная практика подразумевает проверку системы конечным пользователем. Например, пользователь устанавливает актуальную версию VS Code и проверяет качество работы редактора кода или определяет корректность импорта/экспорта тех или иных файлов.
9. **Эксплуатация:** Данная практика подразумевает использование VS Code конечным пользователем. Например, начинающим программистом, который создает свое веб-приложение и используется систему контроля версий в своем проекте.
10. **Обслуживание:** Данная практика подразумевает улучшение работы системы из-за устранения тех или иных ошибок, усовершенствовании тех или иных технологий. Например, добавление поддержки фреймворка React, улучшение технологии написания кода.
11. **Вывод из эксплуатации:** Данная практика подразумевает вывод из эксплуатации, если компания Microsoft изымет исходные коды приложения, например, в виду того, что приложение перестанет пользоваться спросом у пользователей или по причине создания нового приложения. Так же данная практика может быть реализована

конечным потребителем путем удаления дистрибутива программы со своего персонального компьютера.