Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления
Кафедра Интеллектуальных информационных технологий

ОТЧЁТ

по ознакомительной практике

Выполнил: А. М. Пигарев

Студент группы
221703
Проверил: В. В. Голенков

Минск 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель:

Закрепить практические навыки формализации информации в интеллектуальных системах с использованием семантических сетей.

Задачи:

- Построение формализованных фрагментов теории интеллектуальных компьтерных систем и технологий их разработки;
- Построение формальной семантической спецификации библиографических источников, соответствующих указанным выше фрагментам;
- Оформление конкретных предложений по развитию текущей версии Стандарта интеллектуальных компьтерных систем и технологий их разработки

1 ПОСТАНОВКА ЗАДАЧИ

Часть 2 Учебной дисциплины "Представление и обработка информации в интеллектуальных системах"

⇒ *библиографическая ссылка**:

- *Стандарт OSTIS*
- *Материалы конференций OSTIS*
- *Стаьтя на тему "Методы и инструменты 3D-реконструкции изображений лиц"*
  ⇒ *URL**:

[https://dspace.spbu.ru/bitstream/11701/10670/1/zabelina,*eport*.*pdf*]

Вопрос 1 по Части 2 Учебной дисциплины "Представление и обработка информации в интеллектуальных системах"

≔ [Прикладные задачи трехмерной реконструкции]

⇒ *библиографическая ссылка**:

- *Технология комплексной поддержки жизненного цикла семантически совместимых интеллектуальных компьютерных систем нового поколения. Монография.*

2 МЕТОДЫ И СРЕДСТВА РЕАЛИЗАЦИИ OSTIS-СИСТЕМ

Определение требований к платформе и ее функциональности[1]

⇒ *требования и функциональность:**:

{ • *Общее представление*

⇒ *описание**:

[Определение требований к платформе и ее функциональности обеспечивает ясное представление о том, какая платформа должна быть создана и какие задачи она должна выполнять. Это служит основой для дальнейшего проектирования компонентов и модулей платформы.]

⇒ *Требования**:

{ • *Сбор требований*

⇒ *описание**:

[Взаимодействие с заинтересованными сторонами, включая пользователей, заказчиков и другие команды, чтобы понять их потребности и ожидания от платформы. Это может включать проведение собеседований, анализ существующих систем или проведение опросов.]

• *Формулирование требований*

⇒ *описание**:

[Преобразование собранных данных в конкретные требования, которым должна соответствовать платформа. Требования могут быть функциональными (что платформа должна делать) или нефункциональными (какие ограничения и качества должна обладать платформа).]

• *Приоритизация требований*

⇒ *описание**:

[Определение относительной важности каждого требования и их приоритетов. Это позволяет сосредоточить усилия на реализации ключевых функций и обеспечении основных требований.]

• *Разработка сценариев использования*

⇒ *описание**:

[Создание типичных сценариев использования платформы для более полного понимания ее функциональности. Это помогает идентифицировать дополнительные требования и определить, как пользователи будут взаимодействовать с платформой.]

• *Документирование требований*

⇒ *описание**:

[Определение относительной важности каждого требования и их приоритетов. Это позволяет сосредоточить усилия на реализации ключевых функций и обеспечении основных требований.]

}

⇒ *Архитектура и дизайн платформы[2]**:

{ • *Выбор архитектурной модели*

⇒ *описание**:

[Изучение различных архитектурных моделей и их применимость к конкретной платформе. Анализ требований и особенностей платформы для определения наиболее подходящей архитектуры. Выбор между монолитной, микросервисной, распределенной или иной архитектурой в зависи-

мости от потребностей платформы.]

- *Определение требований к платформе:*
  ⇒ *описание**:
  [Сбор и анализ требований, предъявляемых к платформе со стороны пользователей и системных компонентов. Учет функциональных, производительностных, безопасностных и масштабируемых требований. Формулировка конкретных требований, которым должна соответствовать платформа.]
- *Проектирование компонентов и модулей платформы:*
  ⇒ *описание**:
  [Разбиение платформы на логические компоненты и модули в соответствии с требованиями и функциональностью. Определение интерфейсов и взаимодействия между компонентами для обеспечения согласованной работы платформы. Учет принципов разделения ответственности (separation of concerns) и модульности при проектировании компонентов.]
- *Управление зависимостями и взаимодействием между компонентами:*
  ⇒ *описание**:
  [Идентификация зависимостей между компонентами платформы. Определение стратегии управления зависимостями, включая выбор инструментов и подходов. Разработка механизмов взаимодействия и коммуникации между компонентами для обеспечения эффективной работы платформы.]

}

}

⇒ *Инструменты разработки и автоматизации**:

{ • *Выбор инструментов разработки*
  ⇒ *описание**:
  [Изучение различных инструментов разработки, таких как интегрированные среды разработки (IDE), редакторы кода, средства отладки и средства анализа кода. Определение, какие инструменты наилучшим образом соответствуют требованиям и потребностям разработчиков платформы.]
- *Автоматизация процессов*
  ⇒ *описание**:
  [Внедрение автоматизации для развертывания, тестирования и сборки платформы. Использование инструментов, таких как системы непрерывной интеграции (CI/CD), для автоматизации процессов сборки и развертывания новых версий платформы. Автоматическое тестирование позволяет обнаруживать ошибки и обеспечивать надежность и качество платформы.]
- *Системы контроля версий*
  ⇒ *описание**:
  [Использование систем контроля версий, таких как Git, для управления кодом платформы. Система контроля версий позволяет отслеживать изменения в коде, управлять ветвлением и слиянием кода, а также обеспечивает коллаборацию разработчиков и восстановление предыдущих версий кода при необходимости.]
- *Конфигурационное управление*
  ⇒ *описание**:
  [Использование инструментов конфигурационного управления, таких как

Ansible, Chef или Puppet, для управления конфигурацией и развертывания компонентов платформы. Это позволяет автоматизировать установку и настройку платформы на различных окружениях и обеспечивает консистентность конфигурации.]

}

Организация процесса разработки платформы[4]

⇒ *описание**:

[Эффективная организация процесса разработки платформы является ключевым фактором для достижения успеха. В этом пункте рассмотрим методологии разработки, распределение задач и управление командой разработчиков платформы, а также взаимодействие с другими командами и проектами в рамках организации.]

⇒ *Процесс**:

{ • *Методологии разработки*

⇒ *описание**:

[Выбор и применение методологии разработки, наиболее подходящей для platform engineering. Это может быть Agile, Scrum, Kanban или другая методология, которая обеспечит гибкость, сотрудничество и ускоренную разработку. Применение методологии помогает управлять жизненным циклом разработки платформы, устанавливать приоритеты задач и обеспечивать прозрачность и коммуникацию в команде.]

• *Распределение задач и управление командой разработчиков*

⇒ *описание**:

[Определение ролей и ответственностей в команде разработчиков платформы. Распределение задач, планирование и отслеживание прогресса работы. Важно установить эффективные коммуникационные каналы и обеспечить коллаборацию между членами команды. Также следует обеспечить мотивацию и поддержку разработчиков для достижения общих целей.]

• *Взаимодействие с другими командами и проектами*

⇒ *описание**:

[становление эффективного взаимодействия с другими командами и проектами в рамках организации. Это может включать проведение регулярных совещаний, синхронизацию требований и планов, обмен опытом и знаниями. Взаимодействие помогает избежать конфликтов и обеспечивает согласованность и сотрудничество между различными командами в организации.]

}

Управление конфигурацией и версионирование[3]

⇒ *описание**:

[Управление конфигурацией и версионирование являются важными аспектами разработки и поддержки платформы. Они позволяют эффективно управлять изменениями, обеспечивать совместимость и обратную совместимость платформы, а также контролировать конфигурацию компонентов. Ниже представлены ключевые элементы, связанные с управлением конфигурацией и версионированием платформы:]

⇒ *Ключевые элементы**:

{ • *Управление конфигурацией платформы*

⇒ *описание**:

[Определение и управление конфигурацией компонентов платформы, включая управление изменениями, управление конфигурационными элементами и управление версиями. Это включает установление процедур и политик, которые обеспечивают консистентность и стабильность платформы.]

- *Методы версионирования*
  ⇒ *описание**:

    [Применение методов версионирования для идентификации и управления различными версиями платформы. Это может включать применение семантического версионирования, где номер версии состоит из основного, минорного и патч-релиза. Каждое изменение платформы должно быть явно отражено в версионировании.]
- *Управление изменениями*
  ⇒ *описание**:

    [Установление процесса управления изменениями, который определяет, как новые функциональности, исправления ошибок или другие изменения вносятся в платформу. Это может включать оценку и утверждение изменений, использование системы отслеживания задач и контроль за процессом слияния изменений.]
- *Обеспечение совместимости и обратной совместимости*
  ⇒ *описание**:

    [Учет совместимости платформы с различными версиями компонентов, операционных систем, сторонних интеграций и других факторов. Обратная совместимость также является важной, чтобы новые версии платформы могли работать существующими приложениями и интеграциями, минимизируя потенциальные проблемы.]

}

ЗАКЛЮЧЕНИЕ

Я выполнил ознакомительную практику, в рамках которой проанализировал научный текст, выделил его ключевые позиции и идеи и применил SCn-вставки в соответствии с принципами формирования SCn-текста. Кроме того, я закрепил навыки формализации текста и правильного оформления библиографических источников. Результатом выполнения практики стало улучшение моих практических навыков и глубокое понимание рассматриваемой темы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Fitzgerald, B., Stol, K.J., Agerfalk, P.J. (Eds.).(2019). Continuous Software Engineering. Springer.

[2] Bass, L., Clements, P., Kazman, R. (2015). Software Architecture in Practice. Addison-Wesley Professional.

[3] Humble, J., Farley, D. (2010). Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional.

[4] Brown, A.W., Wilson, G.C. (2018). The Architecture of Open Source Applications. Lulu.com.