Подход к разработке системы домашней автоматизации

*Д.А. Ногай**, М.А.Собянин*, А.А.Волков*, А.М.Федрак*, И.Н. Яковина**
** Новосибирский государственный технический университет (НГТУ), Новосибирск, Россия*
*** Инженерный лицей НГТУ, Новосибирск, Россия*

Аннотация: **В статье рассматривается решение задачи управления системой домашней автоматизации, разработанное на основе логики фреймов. Затрагиваются особенности программной реализации предложенного решения, которое реализовывается в рамках концепции «Open Source».**

Ключевые слова: **система домашней автоматизации, фреймы, сценарии**

ВВЕДЕНИЕ

Каждому человеку, живущему в компьютерный век, наверное, хотелось бы прийти домой и почувствовать себя свободным от домашней рутины. Именно поэтому люди стали думать о том, как сделать так, чтобы не только человек заботился о доме, но и дом заботился о человеке. Данная идея воплотилась в реальность, в виде всем известного понятия «Умный дом» . Этот термин зачастую используется совершенно неуместно. Видение концепции «умного дома» - как большой экосистемы, объединяющей множество подсистем, позволило нам разработать свою логическую модель, описывающую процессы, в которые происходят при работе системы домашней автоматизации.

ЦЕЛЬ И ЗАДАЧИ

В ходе разработки метода управления на основе сценарного подхода и логики фреймов для имеющегося программно-аппаратного прототипа системы домашней автоматизации решались следующие *задачи:*

1. Разработка общей модели управления для системы домашней автоматизации.
2. Детальная проработка общей модели для нескольких подсистем на основе логики фреймов.
3. Разработка структурных единиц системы на основе логики фреймов.
4. Реализация варианта сценария как сети фреймов для одной подсистемы.
5. Реализация системы автоматизации, основанной на разработанной логической модели.

1. Особенности предложенного подхода

Логика фреймов

Фреймы были впервые предложены в качестве аппарата для представления знаний М. Минским в 1975 г. Фреймовые модели представляют собой систематизированную в виде единой теории технологическую модель памяти человека и его сознания. Под фреймом понимают минимальные структуры информации, необходимые для представления класса объектов, явлений или процессов.

Фрейм, как правило, представляется в виде сети, состоящей из вершин и дуг (отношений), в которых нижние уровни фрейма заканчиваются слотами (переменными), заполняющимися конкретной информацией при вызове фрейма. Значением слота может быть любая информация: текст, числа, математические соотношения, программы, ссылки на другие фреймы. На наш взгляд, логика фреймов отлично описывает все составляющие умного дома, начиная от формата сообщений, и заканчивая логикой сценариев.Рассмотрим основные опорные точки разработки логики «верхнего» уровня для управления «умного дома» с использованием фреймовых моделей.

При разработке базы данных и базы знаний с применением логики фреймов принято выделять две категории фреймов: **протофреймы** и **фреймы-экземпляры. Протофрейм** представляет себой прототип или шаблон для создания **фреймов-экземпляров.** Рассмотрим на конкретном примере как формируется внутренняя логика нашей системы.

Предложенная система понятий базы знаний системы управления устройствами «умного дома», имеет четыре уровня абстракции (см. *Таблицу* 1):

Таблица *№1.* Иерархия понятий разработанной базы знаний

Уровень абстракции	**Пример**
1.**Элемент системы**	Бра в коридоре
2. **Устройство**	Лампа
3. **Endpoint**	Реле
4. **Активирующее воздействие**	Подать логический сигнал

Элемент системы — это объект, представляющий из себя какое-либо устройство. Занимает высший уровень абстракции.

Устройство — тип устройства. В *Таблице №4* был разобран пример светильника. Главное устройство светильника — Лампа.

Endpoint (англ. - "конечное устройство") — конечный исполнитель. В терминологии системы, именуется «актуатором».

Активирующие воздействие — действие, которое необходимо совершить для изменения

состояния актуатора.

Выделенные уровни позволяют наращивать структуру и функциональность системы, что позволяет претендовать на некоторую универсальность и простоту настройки при функционировании. Рассмотрим, каким образом в предложенной структуре понятий может быть сформирован сценарий работы системы на примере одного из устройств.

Фрейм «Устройство: Лампа» (см. *Таблицу* 2) является примером заполненного значениями протофрейма (шаблона) «Устройство», использующегося для формирования фреймов-экземпляров в базе данных системы.

***Таблица №2*. Пример заполненного протофрейма (шаблона) «Устройство»**

Фрейм « Лампа »		
Идентификатор	**X1**	**7AE5**
Название	**X2**	**«Бра в гостинной»**
Состояние	**X3**	**ВКЛ**
Местоположение	**X4**	**«Гостинная**

Сценарии

Основой модуля принятия решений, выступают **фреймы-сценарии**. Основная функция таких фреймов – объединение нескольких фреймов конкретным логическим условием. Для сборки сценария в системе также будет использован шаблон одного шага, включающий набор следующих параметров: *Идентификатор, Вход, Выход, Условие и Новое_состояние.* Для запуска активирующего сценарий воздействия в системе вводится понятие «сообщение», которое позволяет создать некоторое виртуальное единое информационное пространство, где все устройства объединены одним информационным окружением, и могут взаимодействовать друг с другом. Каждое сообщение в системе должно иметь четыре обязательных поля: *Источник, Адресат, Приоритет и Тип сообщения.* Эти поля описывают то, **от кого** это сообщение, **кому** оно предназначено, какой у него **приоритет**, и **что это за сообщение**. Для описания всех сообщений, введен специальный протофрейм «Сообщение», пример заполнения которого приведен в *Таблице* 3. Общая схема взаимодействия фреймов, иллюстрирующая логику работы сценариев, приведена на *Рис.* 1 и 2 в заполненном определенными параметрами виде.

***Таблица №3*. Фрейм-экземпляр, описывающий пример сообщения**

Фрейм-экземпляр «Результат запланированного измерения»	
Идентификатор отправителя	От кого: **93FE**
Идентификатор адресата	Кому: **AAFF**
Идентификатор типа сообщения	Тип сообщения: **10**
Приоритет сообщения	Приоритет: **C**
Идентификатор endpoint	Endpoint: **E8**
Результат измерения	Результат: **230955**

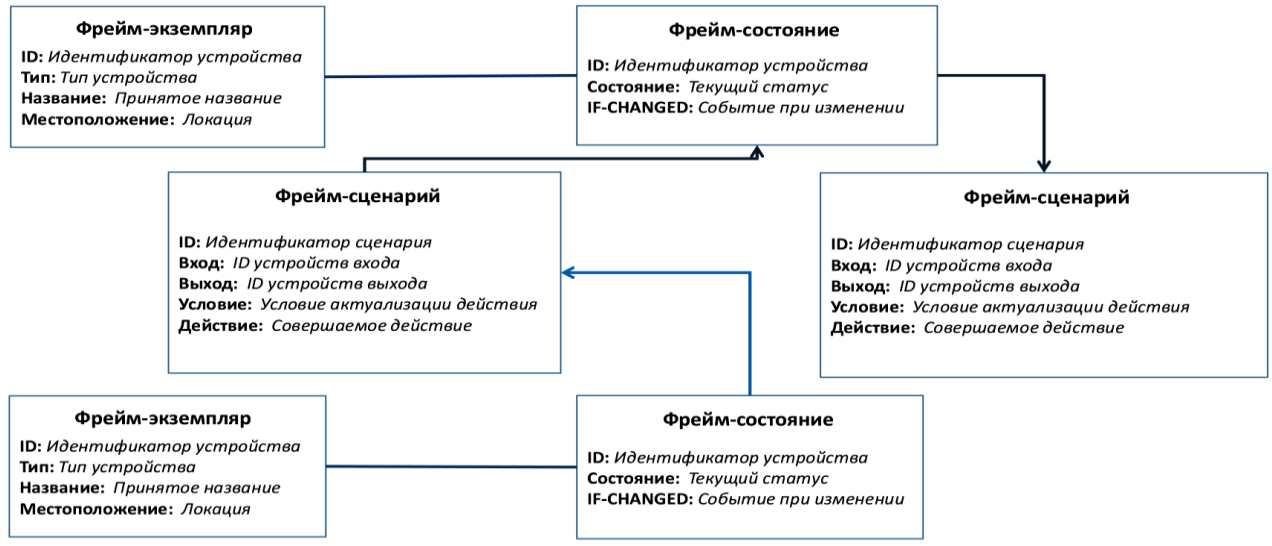

Рис. 1. Схема взаимодействия фреймов

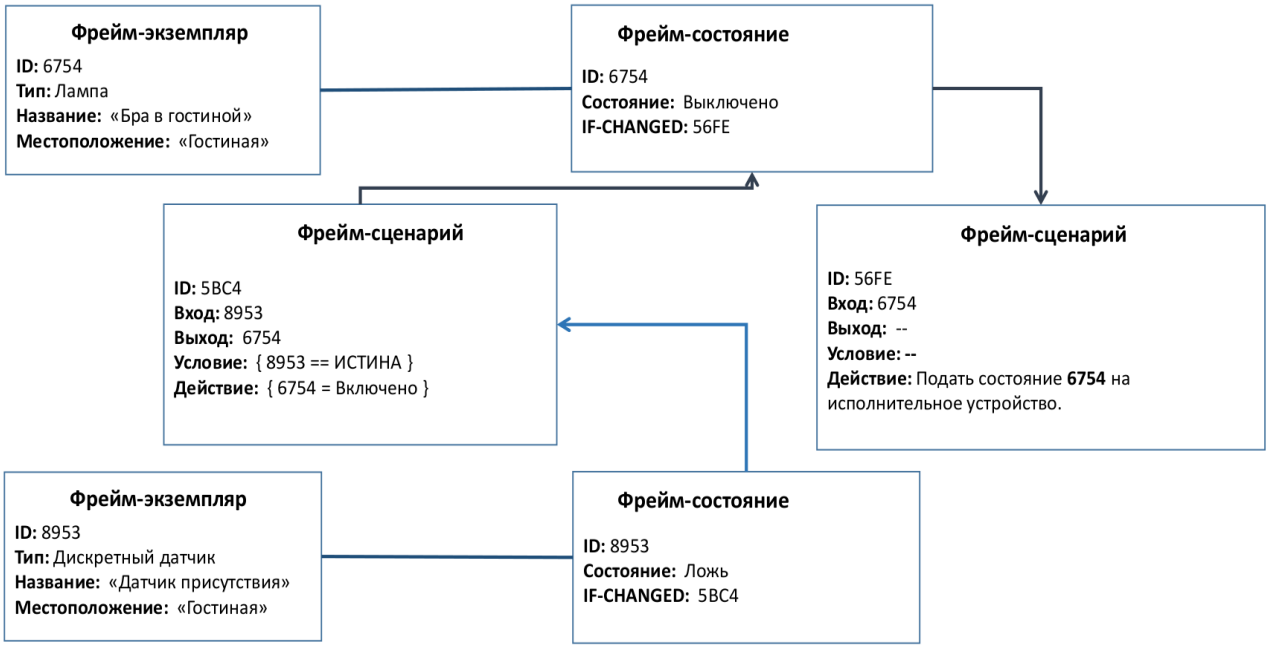

Рис. 2. Схема фреймов с конкретными значениями параметров

Пояснения к *Рис.* 2:

Карта действия сценария является последовательностью следующих действий:

1. Изменение состояния датчика присутствия инициализирует выполнение сценария **5*BC*4.**
2. Фрейм-сценарий **5*BC*4** изменяет фрейм-состояние лампы **6754**.
3. Изменение фрейма-состояния **6754** инициализирует выполнение фрейма-сценария **56*FE***.
4. Фрейм-сценарий 56FE подает команду устройству управления, на изменение физического состояния устройства **5*BC*4**.

2. ОСОБЕННОСТИ РЕАЛИЗАЦИИ ПРЕДЛОЖЕННОГО ПОДХОДА

Устройство управления

Для тестирования разработанного подхода к управлению системой домашней автоматизации и управления низкоуровневой периферией было разработано программное обеспечение для микроконтроллеров архитектуры *AVR* и *ARM*. Оно реализовано на языке C++ с использованием ряда специфичных библиотек для микроконтроллеров. Помимо ядра системы, были написаны две библиотеки, реализующие недостающий в Arduino, функционал:

- **BlackJackString** – библиотека, реализующая альтернативное представление переменных типа String.
- ***ClientCommunications*** – библиотека, реализующая оболочку для протокола обмена данными.

Разработанное для микроконтроллеров, программное обеспечение ***Celestia*** выполняет следующие задачи:

- Получение и отправка сообщений контроллеру и другим устройствам.
- Получение данных с различных датчиков.
- Управление физическими объектами, посредством актуаторов.

Полный функционал *Celestia*:

- ***GPIO*** (*General Purpose Input Output*) – управление портами микроконтроллера.
- **1-*Wire*** – шина данных для коммуникации устройств на малых скоростях в сложных условиях эксплуатации
- ***RS*-232** – стандарт физического уровня для UART (англ. - “Универсальный Асинхронный Приёмо-Передатчик”)
- ***RS*-485** – стандарт физического уровня для UART, применяемый в промышленных системах.
- ***I2C*** – шина данных для использования во встраиваемых системах

MQTT

MQTT (*Message Quene Telemetry Transport*) – протокол прикладного уровня, использующиеся для обмена сообщениями между устройствами по принципу *издатель-подписчик.* Используется для объединения всех устройств системы автоматизации, в единое информационное пространство, по своему принципу, напоминающее своего рода чат-сервер – клиент подключается к *MQTT*-брокеру, и может подписываться на интересующие его каналы, а также передавать свои сообщения в эти или другие каналы. Из всех прочих преимуществ, особенно выделяются следующие:

- Простота реализации
- Протокол опубликован под свободной лицензией

- Низкая нагрузка на канал связи
- Ориентированность на работу в условиях потери связи
- Отсутствие ограничений на формат передаваемых данных
- Возможность интеграции *TLS*-стека для создания защищенных соединений

На *Рис.* 3 приведен пример подключения различных устройств к *MQTT*-брокеру. Все подключенные устройства находятся в едином информационном пространстве, и могут обмениваться данными между собой в режиме реального времени. На *Рис.* 4 приведен пример организации системы автоматизации. В нём присутствуют два устройства управления, разделяющие функционал системы автоматизации на две условные физические зоны – сад и дом, а также присутствует стороннее устройство – смартфон, с которого собираются данные об оповещениях, положении в пространстве (*GPS* и *IMU*) и физическом состоянии владельца (фитнесс-трекер).

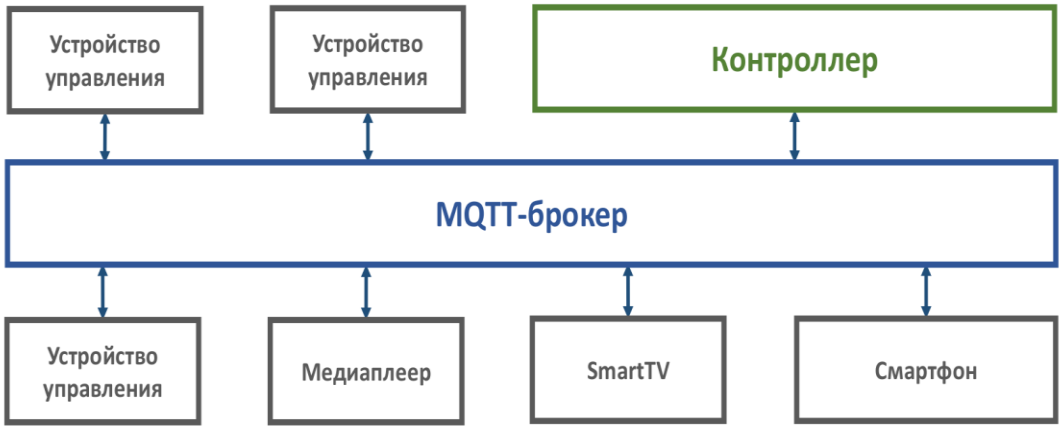

Рис. 3. Пример подключения устройств к *MQTT*-брокеру.

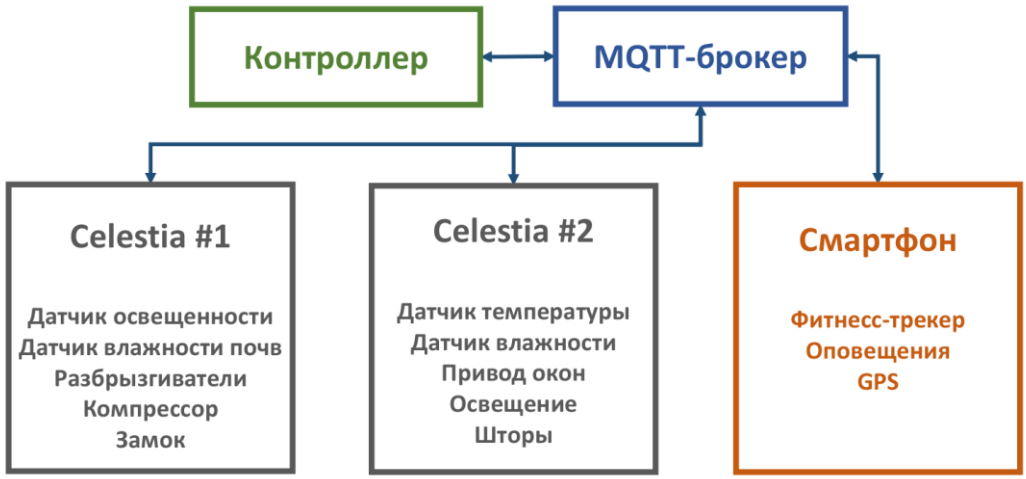

Рис. 4. Пример организации конкретной системы автоматизации

Контроллер

Контроллер в контексте работы представляет комплекс программного обеспечения, реализующий функционал объединения подсистем, мониторинга и принятия решений. Функцию контроллера выполняет ПО, разработанное на языке JavaScript для использования в интерпретаторе Node.js.

Основной задачей контроллера является объединение различных подсистем посредством модуля принятия решений (в случае если действие связано с сценарием) или напрямую (если действие инициализировано через прямой интерфейс).

Внутренние интерфейсы

Внутренние интерфейсы контроллера основаны на логике взаимодействия модулей в Node.js. Каждый подгружаемый модуль получает доступ к различным функциональным компонентам через *модули-фабрики,* а также через *единое модульное пространство* – глобальный объект, содержащий в

себе описания и ссылки на все функции всех подключенных модулей. Также, внутренние интерфейсы общаются между собой посредством модели ***EventEmitter.*** Логика этой модели подразумевает, что определенный модуль может инициализировать события, на которые могут реагировать другие модули. Данный принцип схож с принципом аппаратных прерываний.

Базы данных

Хранилище данных в системе реализовано на основе двух программных продуктов – ***RRDtool*** и ***Loki.JS***. ***RRDtool*** отвечает за получение, хранение и выдачу статистической информации (к примеру, данные датчиков). **Loki.JS** хранит в себе всю системную информацию – профили пользователей, список подключенных устройств, текущий статус различных устройств и тому подобное.

RRDtool – это база данных построенная по приципу *Round-Robin Database* (циклическая база данных). Используется для хранения различных дискретизированных данных с привязкой ко времени. Характеристики и размерность такой базы данных задается единожды, при её создании. Размерность RRD определяется количеством значений за единицу времени.

БД для конкретного датчика можно описать следующим набором условий:

- *Частота поступления данных в базу – 300 секунд*
- *Хранить в базе 12 значений за 1 час*
- *Хранить в базе 24 значения за час за каждые сутки*
- *Хранить в базе 31 суточное значение за каждый месяц*
- *Хранить в базе 3 месячных значения за 3 месяца*
- […]

При создании, база данных принимает свой окончательный размер, и далее, в процессе использования, размер базы данных всегда будет одинаковым.

Loki.js – это база данных, хранящая данные в оперативной памяти и сохраняющая их на диск через заданные интервалы времени. При использовании источника бесперебойного питания, это позволяет в значительной мере экономить ресурс ячеек флеш-памяти, что критически важно во встраиваемых системах. Данные в этой базе хранятся в виде объекта, содержащего коллекции, внутри которых и располагаются данные. При сохранении данных, они сериализуются в формат *JSON*. Данная база данных является встраиваемой (как к примеру, *SQLite*) и не требует наличия отдельного сервера БД.

База знаний

База знаний представляет собой хранилище различных сценариев, абстрагированных от всего, кроме поступающих данных. Концепция данной базы данных подразумевает, что сценарии, хранящиеся в ней не должны иметь никакой абсолютной связи с устройствами системы автоматизации. Они должны представлять себой типичный “чёрный ящик”, получающий на вход некоторый набор данных, и выдающий другой набор данных. Такой подход к абстракции хранилища сценариев, позволяет очень просто создавать и переносить их из одной системы в другую систему.

Модуль принятия решений

Модуль принятия решений отвечает за обработку всей получаемой информации, передачу её в необходимые сценарии, и дальнейшее выполнение этих сценариев. При получении данных, модуль обращается к динамической карте сценариев, определяет сценарии, которым необходимы полученные данные, затем передает полученные данные в базу знаний, получает обработанные данные, и далее выполняет заданные действия.

Мониторинг

Функция постоянной обработки поступающих данных реализуется посредством блока Мониторинг, который является частью *модуля принятия решений,* так как он отвечает за анализ полученных данных и их сравнения с заданными границами значений. Второе предназначение блока мониторинга – отслеживание состояния системы. В случае выхода какого-либо узла или модуля, он сообщает об этом пользователю, и переводит систему в *режим ограниченной функциональности*, перехватывая запросы к вышедшему из строя, узлу. Такой подход позволяет минимизировать ущерб для системы в целом, при выходе из строя части этой системы.

Интерфейсы

REST API

RESTfull API отвечает за прямой доступ к системе, через протокол HTTP. Выполняемые команды передаются в теле запроса, в определенном формате. Данный интерфейс предназначен в первую очередь для M2M (Machine-to-Machine) взаимодействия с иными системами. Ниже, представлен пример API-запроса, целью которого, является запуск определенного сценария с заданными параметрами:

```
GET /api/v1/scenario/run/skeleton.example?temp=34.5
HTTP/1.1
Host: myrhcs.local
Connection: close
X-API-KEY: 4c7a5e51-6ce4-4e2d-97a9-c32532266f0b
```

В представленном выше, HTTP-запросе, передается команда на исполнение сценария, названного «skeleton.example» с аргументом «temp». В заголовках запроса, также передается

параметр «X-API-KEY», хранящий ключ доступа к API. В ответ, сервер передаст информацию о состоянии задания, сериализованную в JSON:

{"code": 200, "execution": "completed"}

В данном случае, сервер сообщает, что запрос успешно принят, и выполнения сценария завершено.

WebSockets API

WebSockets API отвечает за доступ к системе в режиме реального времени. В частности, этот тип API используется *пользовательским интерфейсом* для взаимодействия с системой. Основным отличием от REST API является возможность работы WebSockets API в режиме реального времени, что позволяет удобно взаимодействовать с системой извне.

Пользовательский интерфейс

Основным (на данный момент) способом взаимодействия пользователя с системой, является *пользовательский интерфейс*. Он реализован в виде веб-приложения. Технологический стек веб-интерфейса показан на *рисунке 5*.

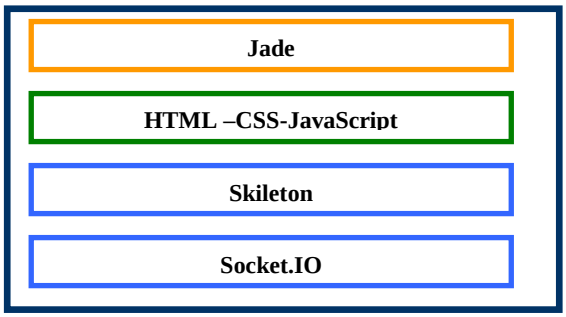

Рис. 5. Стек веб-интерфейса

Jade – это движок для рендеринга HTML-шаблонов. Все страницы веб-интерфейса хранятся в виде набора .jade-файлов, которые рендерятся после каждого их изменения.

Полностью, страница отрисовывается уже в браузере – все необходимые для этого данные (подключенные устройства, их состояния) передаются через *WebSockets*. Это позволяет при необходимости, изменять содержимое страницы, не перезагружая её.

В качестве фреймворка для визуальной составляющей, используется ***Skeleton***. Использование единого фреймворка для визуальной составляющей, позволяет создавать гармоничные интерфейсы, выстроенные в едином стиле. Помимо этого, Skeleton реализует адаптивные сетки, которые позволяют корректно размещать контент веб-страницы как на мобильных устройствах, так и на устройствах с большими диагоналями экранов.

За обработку пользовательского интерфейса на стороне фронтенда, отвечает приложение, написанное на JavaScript, и не требующее дополнительных библиотек, кроме **Socket.io** – библиотеки для работы с WebSockets.

3. ЗАПЛАНИРОВАННЫЙ ФУНКЦИОНАЛ

В данный момент, основным направлением в разработке системы, является доработка базового функционала до полностью работоспособного состояния, с последующим рефакторингом кода (переработкой спорных моментов). После решения этих задач, есть два наиболее важных направления.

Реализация различных способов взаимодействия с системой

На данный момент, в системе присутствует только один способ прямого взаимодействия с пользователем – через веб-интерфейс. В планах задача реализации голосового управления (посредствам CMU Sphinx, или облачного сервиса Yandex.SpeechKit).

Помимо этого, планируется разработка устройств, позволяющих физически взаимодействовать с системой (миниатюрные сенсорные панели).

Ещё одним перспективным способом взаимодействия с пользователем, является распознавание жестов, но в этой сфере существует ряд концептуальных проблем (трудность визуального распознавания жестов на больших площадях).

Интеграция сторонних сервисов обработки данных

Для улучшения интеллектуальных способностей системы, планируется добавить поддержку облачной платформы *IBM Bluemix* в систему автоматизации. Данная платформа даёт доступ к системе *IBM DeepQA,* используемой суперкомпьютером *IBM Watson*. Из всех технологий, предоставляемых *IBM Bluemix*, в рамках решения поставленных задач, наиболее перспективными являются следующие:

- *AlchemyLaunguage* – система, которая позволяет распознавать естественную речь, раскладывая её на определённые логические элементы.
- *AlcemyData* – система, позволяющая обрабатывать данные на естественном языке из различных источников (новостные сайты, блоги и т. д.), получая к ним доступ в виде базы данных.
- *AlchemyVision* – система обнаружения и классификации различных объектов на изображении.
- *Dialog* – сервис, позволяющий строить диалоги между пользователем и системой на естественном языке.
- *Personality Insights* – система, позволяющая на основе разнородных данных о пользователе, строить его психологический и

эмоциональный портрет.

- *Natural Launguage Classifier* – сервис, предоставляющий наиболее подходящие варианты реакции на запрос, сформулированный на естественном языке.

4. ЗАКЛЮЧЕНИЕ

В ходе работы над проектом, в рамках сформулированной цели: разработка метода управления «умным домом» на основе сценарного подхода и логики фреймов для имеющегося программно-аппаратного прототипа системы домашней автоматизации, были решены следующие задачи:

- Разработана общая модель управления для разработанного ранее прототипа системы домашней автоматизации с учетом сложившихся для нее требований: отказоустойчивость, иерархичность, гибкость и расширяемость.
- Рассмотрены особенности использования фреймовых моделей для решения задачи разработки метода управления «умным домом».
- Для нескольких подсистем «умного дома» на основе «логики фреймов» детально проработана общая модель управления.
- Выполнена реализация варианта сценария как сети фреймов для одной подсистемы.
- Выявлены проблемы и предложены пути их решения.

Распространение исходного кода

Система автоматизации, представленная в данной работе, разрабатывается под открытой лицензий в рамках концепции «Open Source». Все исходные коды лицензированы под лицензией MIT (если в конкретном случае не указано иное), и находятся в публичном доступе, в Git-репозиториях по адресам:
https://github.com/skbrii/RHCS/tree/celestia
https://github.com/skbrii/RHCS/tree/OctopusAdventure

ЛИТЕРАТУРА

[1] *Макаренко С.И.* Интеллектуальные информационные системы / Ставропольский филиал ГОУ ВПО «Московский государственный гуманитарный ун-т » – Ставрополь, 2009. – 70 с.

[2] *Ногай Д.А., Богомолов М.А., Чурсин Г.А.* Разработка системы управления цифровыми и аналоговыми устройствами дома // Национальное Достояние России 2015: материалы науч. конф. – Новосибирск – 1 - 35 с.

[3] Мой удобный дом [Электронный ресурс] // Гиктаймс – 2014. – URL: http://geektimes.ru/post/258346/ (дата обращения: 13.11.2015).

[4] Контроллер центральный домашний, всемогущий КЦД-В-2-12 [Электронный ресурс] // Гиктаймс – 2014. – URL: http://geektimes.ru/post/258350/ (дата обращения: 13.11.2015).

[5] Прибор измерительный температуры и влажности ПИ-ТВ-2 [Электронный ресурс] // Гиктаймс – 2014. – URL: http://geektimes.ru/post/258358/ (дата обращения: 13.11.2015).

[6] Автомат света и музыки АСИМ-АУ-2-6 [Электронный ресурс] // Гиктаймс – 2014. – URL: http://geektimes.ru/post/258390/ (дата обращения: 13.11.2015).

[7] *Andrey_B.* Микросети 1-wire [Электронный ресурс] // Ab-log.ru – 2008. – URL: http://ab-log.ru/smart-house/1-wire (дата обращения: 13.11.2015).

[8] *Andrey_B.* Моя сеть 1-wire. Опыт и рекомендации [Электронный ресурс] // Ab-log.ru – 2010. – URL: http://ab-log.ru/smart-house/1-wire/1-wire-experience (дата обращения: 13.11.2015).

[9] *Andrey_B.* Как я делал уличный датчик освещенности [Электронный ресурс] // Ab-log.ru – 2013. – URL: http://ab-log.ru/smart-house/ethernet/light-sensor-outdoor (дата обращения: 13.11.2015).

[10] *Andrey_B.* MegaD-328 - Готовый многофункциональный контроллер Умного Дома [Электронный ресурс] // Ab-log.ru – 2012. – URL: http://ab-log.ru/smart-house/ethernet/megad-328 (дата обращения: 13.11.2015).

[11] Фреймы: основные понятия и вопросы практического применения [Электронный ресурс] // 2013. – URL: http://edu.dvgups.ru/METDOC/EKMEN/MEN/SIST_UPR/METOD/OSN_UPR/Klykov_25.htm (дата обращения: 10.11.2015)

Д.А. Ногай – учащийся Инженерного лицея, НГТУ, Новосибирск, Россия

М.А. Собянин – студент НГТУ, Новосибирск, Россия

А.А. Волков – студент НГТУ, Новосибирск, Россия

А.М. Федрак – студент НГТУ, Новосибирск, Россия

Яковина Ирина Николаевна – к.т.н., доцент кафедры вычислительной техники НГТУ.
Автор более 40 научных работ. Сфера научных интересов: методы интеллектуального анализа данных, интеллектуальные системы, робототехника.
E-mail: irina.nir@gmail.com

The approach to the development of home automation system

D.A.Nogai M.A.Sobyanin, A.A. Volkov, A.M.Fedrak, I.N. Yakovina

REFERENCES

[1] Makarenko S.I. Intellektual'nye informacionnye sistemy / Stavropol'skij filial GOU VPO «Moskovskij gosudarstvennyj gumanitarnyj un-t » – Stavropol', 2009. – 70 s.

[2] Nogaj D.A., Bogomolov M.A., Chursin G.A. Razrabotka sistemy upravlenija cifrovymi i analogovymi ustrojstvami doma // Nacional'noe Dostojanie Rossii 2015: materialy nauch. konf. – Novosibirsk – 1 - 35 s.
[3] Moj udobnyj dom [Jelektronnyj resurs] // Giktajms – 2014. – URL: http://geektimes.ru/post/258346/ (13.11.2015).
[4] Kontroller central'nyj domashnij, vsemogushhij KCD-V-2-12 [Jelektronnyj resurs] // Giktajms – 2014. – URL: http://geektimes.ru/post/258350/ (13.11.2015).
[5] Pribor izmeritel'nyj temperatury i vlazhnosti PI-TV-2 [Jelektronnyj resurs] // Giktajms – 2014. – URL: http://geektimes.ru/post/258358/ (13.11.2015).
[6] Avtomat sveta i muzyki ASIM-AU-2-6 [Jelektronnyj resurs] // Giktajms – 2014. – URL: http://geektimes.ru/post/258390/ (13.11.2015).
[7] Andrey_B. Mikroseti 1-wire [Jelektronnyj resurs] // Ab-log.ru – 2008. – URL: http://ab-log.ru/smart-house/1-wire (13.11.2015).
[8] Andrey_B. Moja set' 1-wire. Opyt i rekomendacii [Jelektronnyj resurs] // Ab-log.ru – 2010. – URL: http://ab-log.ru/smart-house/1-wire/1-wire-experience (13.11.2015).
[9] Andrey_B. Kak ja delal ulichnyj datchik osveshhennosti [Jelektronnyj resurs] // Ab-log.ru – 2013. – URL: http://ab-log.ru/smart-house/ethernet/light-sensor-outdoor (13.11.2015).
[10] Andrey_B. MegaD-328 - Gotovyj mnogofunkcional'nyj kontroller Umnogo Doma [Jelektronnyj resurs] // Ab-log.ru – 2012. – URL: http://ab-log.ru/smart-house/ethernet/megad-328 (13.11.2015).
[11] Frejmy: osnovnye ponjatija i voprosy prakticheskogo primenenija [Jelektronnyj resurs] // 2013. – URL: http://edu.dvgups.ru/METDOC/EKMEN/MEN/SIST_UPR/METOD/OSN_UPR/Klykov_25.htm (10.11.2015)