ISSN 1694-5050

НАЦИОНАЛЬНАЯ АКАДЕМИЯ НАУК КЫРГЫЗСКОЙ РЕСПУБЛИКИ

ИНСТИТУТ МАШИНОВЕДЕНИЯ И АВТОМАТИКИ

ПРОБЛЕМЫ АВТОМАТИКИ И УПРАВЛЕНИЯ

№ 1 (38)

Журнал входит в РИНЦ, а также в перечень
ВАК Кыргызской Республики рецензируемых
научных журналов, в которых должны быть опубликованы
основные результаты диссертаций на соискание
ученых степеней доктора и кандидата наук

Импакт-фактор – 0,256

Журнал издается с 1996 г.

Зарегистрирован Министерством юстиции
Кыргызской Республики
Свидетельство № 1503 от 24 марта 2020 г.

БИШКЕК ***2020*** **ИЛИМ**

П 78

Проблемы автоматики и управления: Научно-технический журнал
/ Национальная академия наук Кыргызской Республики.–
Бишкек: Илим, 2020.– №1 (38).– 110 с.

В журнале публикуются статьи по оптимизации динамических систем, моделированию и программному обеспечению, информационно-измерительным системам, а также системам автоматики и диагностики

Контакты редакции:
720071, Бишкек, проспект Чуй, 265.
Институт машиноведения и автоматики Национальной академии наук КР

Телефон: +996 312 39-20-36.
E-mail: avtomatika_nankr@mail.ru
Полная электронная версия журнала: https://ima.naskr.kg/jrn/index.php/pau

ISSN 1694-5050

СОДЕРЖАНИЕ

УПРАВЛЕНИЕ И МОДЕЛИРОВАНИЕ ДИНАМИЧЕСКИХ СИСТЕМ И ПРОЦЕССОВ

УДК 62-50 10.5281/zenodo.3904077

Ж. Шаршеналиев
Институт машиноведения и автоматики НАН КР, г.Бишкек

О ПОДХОДАХ УПРОЩЕНИЯ СЛОЖНЫХ ДИНАМИЧЕСКИХ СИСТЕМ

Изложены приемы исследования сложных объектов, системный подход, взаимодействие и взаимосодействие.

Ключевые слова: общая теория систем, системный подход; декомпозиция; агрегирование; обратная связь; взаимосодействие.

«Системой можно назвать только такой комплекс избирательно вовлеченных компонентов, у которых взаимодействия и взаимоотношения принимают характер взаимоСОдействия компонентов на получение фокусированного полезного результата».

П.К. Анохин. Философская Энциклопедия. II Гл. ред. Ф.В. Константинов. – М. Сов. энциклопедия, 1962. Т. 2. – 375 с.

В настоящее время системный подход стал доминирующим подходом в общей теории систем. При этом при исследовании сложных объектов системный подход является новой методологией, учитывающей не только их взаимодействия, но и взаимоСОдействия.

В рамках системного подхода актуальным является использование методов упрощения – декомпозиции, агрегирования, трансформации и сингулярности.

1. *Декомпозиция* – расслоение сложной системы или объекта на несколько более простых независимых частей.

2. *Агрегирование* – объединение нескольких простейших частей в более сложные укрупненные с целью ограничения чрезмерного повышения порядка. Агрегированная система есть новая форма укрупненных переменных, т.е. агрегатов.

Необходимо отметить, что декомпозиция и агрегирование составляют основные способы исследования сложных динамических систем.

3. *Трансформация* – преобразование, не изменяющее порядка математической модели, но приводящее ее к более удобному виду.

4. *Сингулярность* – наличие в математической модели объекта тех или иных неправильностей по сравнению с регулярными объектами того же рода.

Обычно уравнения динамической системы управления описываются уравнениями первого порядка относительно каждой из переменных состояния. В общем случае уравнения динамики запишем в виде:

$$\begin{aligned}
&\dot{x}_1 = a_{11}x_1 + a_{12}x_2 + \cdots a_{1n}x_n + b_{11}u_1 + \cdots + b_{1m}u_m,\\
&\dot{x}_2 = a_{21}x_1 + a_{22}x_2 + \cdots a_{2n}x_n + b_{21}u_1 + \cdots + b_{2m}u_m,\\
&\vdots\\
&\dot{x}_n = a_{n1}x_1 + a_{n2}x_2 + \cdots a_{nn}x_n + b_{n1}u_1 + \cdots + b_{nm}u_m.
\end{aligned} \tag{1}$$

Эту систему уравнений представим в матричной форме:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}a_{12}\ldots a_{1n} \\ a_{21}a_{22}\cdots a_{2n} \\ \vdots \\ a_{n1}a_{n2}\ldots a_{nn} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_{11}\cdots b_{1m} \\ \vdots \\ b_{n1}\cdots b_{nm} \end{bmatrix}\begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix}. \tag{2}$$

В компактном в виде (2) имеет следующий вид как уравнение состояния:

$$\dot{X} = AX + BU, \tag{3}$$

где $X \in R^n$, $u \in R^m$; A – постоянная $n \times n$, B – постоянная $n \times m$ матрицы.

Разбиваем матрицу A на блоки так, чтобы диагональные блоки были квадратными:

$$\dot{X} = \begin{bmatrix} A_1A_{12}A_{13} \\ A_{21}A_2A_{23} \\ A_{31}A_{32}A_3 \end{bmatrix}X + \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}U. \tag{4}$$

Считая, что переменные состояния, множителями при которых являются элементы матрицы A_i $(i-го блока)$, относятся к вектору состояния X_i подсистемы S_i, разобьем уравнение (3) на S подсистем

$$\dot{X}_i = A_i x_i + \sum_{\substack{i=1 \\ i \neq j}}^{S} A_{ij}x_j + B_i u_i, i = 1,2,\cdots S, \tag{5}$$

где $X_i \in R^{n_i}$; матрица A_i отображает собственные динамические свойства подсистемы S_i; выражение $\sum A_{ij}x_j$, содержащее все остальные переменные x_j, кроме собственного вектора x_i подсистемы S_i, характеризует связи между подсистемами; матрица A_{ij} есть матрица связей.

Такая трансформация характеризуется тем, что ни одна из компонент вектора x_i не является одновременно компонентой какого-либо другого вектора x_j другой подсистемы S_j. Такие подсистемы называют не перекрывающимися.

Теперь если матрица A линейной системы расчленена на подсистемные блочные матрицы, собственно, подсистемы A_i, то вместо матриц связей A_{ij} в (5) вводим матрицы $\varepsilon_i A_{ij}$. Здесь малый параметр ε_i учитывает слабость связей. Допуская, что $\varepsilon_j = 0,\ i \neq j$, имеем изолированные подсистемы. При $\varepsilon_j = 0$ такое обращение называется **структурным возмущением.**

Необходимо отметить, что при структурном возмущении часто в качестве множителя при коэффициентах связей вводят величину $\varepsilon(t)$ вместо ε, которая может принимать значения $0 \le \varepsilon(t) \le 1$. Задаваемое извне произвольное изменение $\varepsilon(t)$ есть **параметрическое возмущение**. В отличие от параметрического возмущения структурные возмущения **регулярны**. А в случае сильных связей коэффициенты матриц A_i и A_{ij} являются соизмеримыми друг с другом, т.е. $a_{ij} \rangle\rangle 1,\ \ i, j = 1,2,\cdots,\tau$.

Тогда величину a_{ij} представим в виде $a_{jj} = K\overline{a}_{ij}$, и малый параметр ε обозначим как $\varepsilon = K^{-1}$. Тогда уравнение соответствующей подсистемы представим в виде

$$\varepsilon \dot{X}_i = \overline{A}_i x_i + \sum_{\substack{i=1 \\ i \neq l}}^{\tau} \overline{A}_{ij} x_j,\ i = 1,2,\cdots,\tau. \qquad (6)$$

Учитывая, что изменение ε от конечной величины «1» до нуля или от нуля до конечной величины «1» приводит к изменению порядка уравнения. Такие возмущения являются **сингулярными.** При исследовании таких систем выдвигается проблема начальных значений.

Одним из известных методов декомпозиции (понижения порядка) уравнения есть простое отбрасывание малых членов с производными высших порядков.

Математические обоснования способов понижения порядка математических моделей в динамических системах управления были разработаны М.В. Мееровым [1] и А.Н. Тихоновым и их научной школой.

Литература

1. Мееров М.В. Синтез структур систем автоматического регулирования высокой точности. М., 1959.

УДК 519.3:62–50 10.5281/zenodo.3904081

Т.П. Самохвалова, *sam_tp@mail.ru*
Институт машиноведения и автоматики НАН КР

АЛГОРИТМ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ С ОБРАТНОЙ СВЯЗЬЮ ДЛЯ ПРОЦЕССА С ПОСЛЕДЕЙСТВИЕМ

Построен алгоритм управления с обратной связью методом Р. Беллмана для одномерного процесса с последействием.

Ключевые слова: оптимальное управление; обратная связь; принцип оптимальности Р. Беллмана; интегро-дифференциальное уравнение типа Вольтерра; производная по направлению.

Введение

Математические модели с интегро-дифференциальными уравнениями в обыкновенных производных представлены в известной монографии Я.В. Быкова [1] и его школы [2] и др. Задачи оптимального управления с такими моделями не выявлены и не являются изученными. В процедурах решения этих задач требуется дифференцировать по функции функционал. В данной статье разработаны расчетные формулы решения линейных одномерных задач оптимального управления с интегро-дифференциальной моделью с использованием производной по направлению [3, 4]. Алгоритмы оптимального управления построены методом динамического программирования Р. Беллмана, изложенным для дифференциальных моделей в обыкновенных производных в [5, 6].

Приведем определение из [3]. «Известно, что для вещественных функций $F(x)$ одного вещественного переменного x два определения – *существование* конечного предела

$$1)\quad \lim_{h\to 0}\frac{F(\hat{x}+h)-F(\hat{x})}{h}$$

и *возможность* асимптотического разложения при $h\to 0$

$$2)\quad F(\hat{x}+h)=F(\hat{x})+F'(\hat{x})h+o(h)$$

– приводят к одному и тому же понятию дифференцируемости.

Для функций нескольких переменных, а тем более для функций с бесконечномерной областью определения дело обстоит не так просто.

Определение 1) и обобщающее его определение частной производной приводят к понятию *производной по направлению*, первой вариации и производной Гато. В то же время обобщение определения 2) приводит к производной Фреше и строгой дифференцируемости.

Пусть X и Y – линейные нормированные пространства, U – окрестность точки $\hat{x}$ в X, F – отображение из U в Y.

Определение 1. Предел

$$3)\quad \lim_{\alpha\to+0}\frac{F(\hat{x}+\alpha h)-F(\hat{x})}{\alpha}$$

в предположении, что он существует, называется *производной F в точке $\hat{x}$ по направлению h* и обозначается $F'(\hat{x};h)$ (а также у других авторов $D_hF(\hat{x})$ и др.).

Для вещественных функций $(Y = R)$ будем понимать 1) несколько расширенно, допуская в качестве предела $-\infty$ и $+\infty$».

В данной работе будем ориентироваться на определение 1 и на формулу 3) вычисления производной по направлению в тех ситуациях, когда требуется вычислить производную по функции (производную сложной функции, производную под знаком интеграла) [3, 4]. Направление h будем полагать равным единичной функции $h(t) \equiv 1$.

1. Постановка задачи

Рассмотрим задачу оптимального управления одномерным процессом, моделью которого является линейное интегро-дифференциальное уравнение типа Вольтерра

$$\frac{dx(t)}{dt} = Ax(t) + \lambda \int_0^t G(t,\tau)x(\tau)d\tau + Bu(t), \quad x(0) = x_0, \quad t \in [0;T]. \tag{1}$$

Минимизируемый критерий качества запишем в виде

$$J = \gamma_1 \int_0^T Qx^2(t)dt + \gamma_2 Fx^2(T) + \beta \int_0^T u^2(t)dt. \tag{2}$$

Здесь A, B, λ, x_0 – заданные постоянные, $T, \gamma_1, \gamma_2, Q, F, \beta$ – положительные постоянные, $G(t,\tau)$ – заданная непрерывная дифференцируемая по t функция; $x(t)$ – абсолютно непрерывная функция; $u(t)$ – управляющая функция из множества W допустимых управлений. На рис. 1 приведен график состояния процесса $x(t)$ при нулевом управлении $u(t) \equiv 0$. В критерии (2) требуется минимизация отклонения состояния процесса $x(t)$ от нуля.

2. Решение задачи 1

Задача 1. Найти управление $u(t, x(t)) \in W$ и соответствующее решение $x(t)$ уравнения (1), минимизирующие критерий (2).

Для решения задачи 1 применим метод динамического программирования Р.Беллмана, разработав модификации, связанные с интегральным слагаемым в модели (1). Обозначим

$$y_1(x) = \int_0^t G(t,\tau)x(\tau)d\tau, \ \frac{d}{dx}y_1(x) = \int_0^t G(t,\tau)d\tau = v(t). \tag{3}$$

Выполнив соответствующие преобразования, следуя [5, 6], учитывая обозначения (3), получим для (1), (2) функциональное уравнение Беллмана в виде

$$-\frac{\partial S(t,x)}{\partial t} = \gamma_1 Qx^2(t) + Ax\frac{\partial S(t,x)}{\partial x} + \lambda \int_0^t G(t,\tau)x(\tau)d\tau \cdot \frac{\partial S(t,x)}{\partial x} - \frac{B^2}{4\beta}\left(\frac{\partial S(t,x)}{\partial x}\right)^2 \tag{4}$$

с условием в конечный момент времени

$$S(T,x) = \gamma_2 Fx^2(T). \tag{5}$$

В данной статье решение уравнения (4) предлагается искать в виде формы второго порядка

$$S(t,x(t)) = K(t)x^2(t) + \lambda K_1(t)(y_1(x))^2 + \lambda K_2(t)x(t)y_1(x),$$

$$\text{или} \quad S(t,x(t)) = K(t)x^2(t) + \lambda K_1(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)^2 + \lambda K_2(t)x(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right), \tag{6}$$

где $K(t), \ K_1(t), \ K_2(t)$ – пока не определенные вспомогательные функции.

С учетом (6) и формулы Ньютона-Лейбница дифференцирования интеграла по параметру выпишем левую часть $-\dfrac{\partial S(t,x)}{\partial t}$ уравнения Беллмана (4), состоящую из выражения

$$-\frac{\partial S(t,x)}{\partial t}=-\frac{dK(t)}{dt}x^2-\lambda\frac{dK_1(t)}{dt}\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)^2-$$

$$-2\lambda K_1(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)\cdot\left(\int_0^t \frac{\partial}{\partial t}G(t,\tau)x(\tau)d\tau+G(t,t)x(t)\right)-$$

$$-\lambda\frac{dK_2(t)}{dt}x\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)-\lambda K_2(t)x\left(\int_0^t \frac{\partial}{\partial t}G(t,\tau)x(\tau)d\tau+G(t,t)x(t)\right). \qquad (7)$$

Правую часть уравнения Беллмана (4) здесь не выписываем, так как для (6) она весьма громоздкая. Группируя коэффициенты при линейно независимых функциях $x^2(t)$, $(y_1(t))^2$, $x(t)y_1(t)$, выпишем полученные средствами пакета Maple уравнения относительно $K(t)$, $K_1(t)$, $K_2(t)$, конечные условия определим по (5). Получим систему типа Риккати для задачи 1:

$$-\frac{dK(t)}{dt}=\gamma_1 Q+2AK(t)-\frac{B^2}{\beta}K^2(t)-\lambda\frac{B^2}{\beta}K(t)K_2(t)v(t)+\lambda AK_2(t)v(t)-$$

$$-\lambda^2\frac{B^2}{4\beta}K_2^2(t)v^2(t)+\lambda K_2(t)G(t,t),\quad K(T)=\gamma_2 F; \qquad (8)$$

$$-\lambda\frac{dK_1(t)}{dt}=-\lambda^2\frac{B^2}{\beta}K_1^2(t)v^2(t)-\lambda^2\frac{B^2}{\beta}K_1(t)K_2(t)v(t)+$$

$$+2\lambda^2K_1(t)v(t)-\lambda^2\frac{B^2}{4\beta}K_2^2(t)+\lambda^2K_2(t),\qquad K_1(T)=0; \qquad (9)$$

$$-\lambda\frac{dK_2(t)}{dt}=-\lambda^2\frac{B^2}{2\beta}K_2^2(t)v(t)+2\lambda K(t)+2\lambda AK_1(t)v(t)+\lambda^2K_2(t)v(t)-$$

$$-2\lambda\frac{B^2}{\beta}K(t)K_1(t)v(t)-\lambda\frac{B^2}{\beta}K(t)K_2(t)+\lambda AK_2(t)-$$

$$-\lambda^2\frac{B^2}{\beta}K_1(t)K_2(t)v^2(t)+2\lambda K_1(t)G(t,t),\qquad K_2(T)=0. \qquad (10)$$

В уравнениях (8) – (10) $\lambda\neq 0$, в левой части (7) на подынтегральную функцию $G(t,\tau)$ наложено ограничение

$$\frac{\partial}{\partial t}G(t,\tau)=0. \qquad (11)$$

Оптимальное управление вычисляем по известной формуле [5, 6]

$$u^0(t,x)=-\frac{B}{2\beta}\frac{\partial S(t,x)}{\partial x}, \qquad (12)$$

получаем в задаче 1

$$u^0(t,x)=-\frac{B}{2\beta}\left\{2K(t)x(t)+2\lambda K_1(t)y_1(x)v(t)+\lambda K_2(t)y_1(x)+\lambda K_2(t)x(t)v(t)\right\},$$

или $$u^0(t,x)=-\frac{B}{2\beta}\left\{2K(t)x(t)+2\lambda K_1(t)\int_0^t G(t,\tau)x(\tau)d\tau\int_0^t G(t,\tau)d\tau+\right.$$

$$+\lambda K_2(t)\int_0^t G(t,\tau)x(\tau)d\tau+\lambda K_2(t)x(t)\int_0^t G(t,\tau)d\tau\Big\}. \qquad (13)$$

Из (13) получим расчетную величину управления в первой точке $t=0$:

$$u^0(0,x(0))=-\frac{B}{2\beta}\{2K(0)x(0)\}.$$

3. Решение задачи 2

Если требуется минимизировать отклонение $x(t)$ от ненулевой заданной величины $g\neq 0$, то критерий J записывается в виде

$$J_1=\gamma_1\int_0^T Q(x(t)-g)^2dt+\gamma_2F(x(T)-g)^2+\beta\int_0^T u^2(t)dt. \qquad (14)$$

Задача 2. Найти управление $u(t,x(t))\in W$ и соответствующее решение $x(t)$ уравнения (1), минимизирующие критерий (14).

В задаче 2 уравнение Беллмана имеет вид

$$-\frac{\partial S(t,x)}{\partial t}=\gamma_1Q\big(x(t)-g\big)^2+Ax\frac{\partial S(t,x)}{\partial x}+\lambda\int_0^t G(t,\tau)x(\tau)d\tau\cdot\frac{\partial S(t,x)}{\partial x}-\frac{B^2}{4\beta}\left(\frac{\partial S(t,x)}{\partial x}\right)^2 \qquad (15)$$

с условием в конечный момент времени

$$S(T,x)=\gamma_2F\big(x(T)-g\big)^2. \qquad (16)$$

Соответствующий функционал Беллмана $S(t,x)$ в (15) в задаче 2 предлагается определять в виде формы второго порядка

$$S(t,x(t))=K(t)x^2(t)+\lambda K_1(t)(y_1(x))^2+\lambda K_2(t)x(t)y_1(x)+\varphi(t)x+\lambda\varphi_1(t)y_1(x)+\eta(t),$$

или с учетом обозначений (3)

$$S(t,x(t))=K(t)x^2(t)+\lambda K_1(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)^2+\lambda K_2(t)x(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)+$$

$$+\varphi(t)x+\lambda\varphi_1(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)+\eta(t), \qquad (17)$$

где $K(t)$, $K_1(t)$, $K_2(t)$, $\varphi(t)$, $\varphi_1(t)$, $\eta(t)$ пока не определенные вспомогательные функции. Тогда

$$\frac{\partial S(t,x)}{\partial x}=2K(t)x+2\lambda K_1(t)y_1(x)v(t)+\lambda K_2(t)y_1(x)+2K_2(t)xv(t)+\varphi(t)+\lambda\varphi_1(t)v(t)$$

и оптимальное управление $u^0(t,x)$ в соответствии с (12) определяется в виде

$$u^0(t,x)=-\frac{B}{2\beta}\Big\{2K(t)x(t)+2\lambda K_1(t)y_1(x)v(t)+\lambda K_2(t)[y_1(x)+x(t)v(t)]+\varphi(t)+\lambda\varphi_1(t)v\Big\}$$

или $$u^0(t,x)=-\frac{B}{2\beta}\Big\{2K(t)x(t)+2\lambda K_1(t)\int_0^t G(t,\tau)x(\tau)d\tau\int_0^t G(t,\tau)d\tau+$$

$$+\lambda K_2(t)\int_0^t G(t,\tau)x(\tau)d\tau+\lambda K_2(t)x(t)\int_0^t G(t,\tau)d\tau+\varphi(t)+\lambda\varphi_1(t)\int_0^t G(t,\tau)d\tau\Big\}. \qquad (18)$$

Из (18) получим расчетную величину управления в первой точке $t=0$:

$$u^0(0,x(0))=-\frac{B}{2\beta}\{2K(0)x(0)+\varphi(0)\}.$$

Левая часть $-\frac{\partial S(t,x)}{\partial t}$ уравнения Беллмана (15) в задаче 2 дополняется слагаемыми относительно функций $\varphi(t),\ \varphi_1(t),\ \eta(t)$ и принимает вид:

$$-\frac{\partial S(t,x)}{\partial t}=-\frac{dK(t)}{dt}x^2-\lambda\frac{dK_1(t)}{dt}\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)^2-$$
$$-2\lambda K_1(t)\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)\cdot\left(\int_0^t \frac{\partial}{\partial t}G(t,\tau)x(\tau)d\tau+G(t,t)x(t)\right)-$$
$$-\lambda\frac{dK_2(t)}{dt}x\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)-\lambda K_2(t)x\left(\int_0^t \frac{\partial}{\partial t}G(t,\tau)x(\tau)d\tau+G(t,t)x(t)\right)-$$
$$-\frac{d\varphi(t)}{dt}\cdot x-\lambda\frac{d\varphi_1(t)}{dt}\left(\int_0^t G(t,\tau)x(\tau)d\tau\right)-$$
$$-\lambda\varphi_1(t)\left(\int_0^t \frac{\partial}{\partial t}G(t,\tau)x(\tau)d\tau+G(t,t)x(t)\right)-\frac{d\eta(t)}{dt}. \qquad (19)$$

Правую часть уравнения Беллмана в задаче 2 также не выписываем, так как для (15) она еще более громоздкая. Группируем коэффициенты при линейно независимых функциях $x^2(t),\ (y_1(t))^2,\ x(t)y_1(t),\ x(t),\ y_1(t)$ и свободные члены, выпишем уравнения относительно $K(t),\ K_1(t),\ K_2(t),\ \varphi(t),\ \varphi_1(t),\ \eta(t)$. В задаче 2 уравнения относительно $K(t),\ K_1(t),\ K_2(t)$ совпадают с (8) – (10). Уравнения относительно $\varphi(t),\ \varphi_1(t),\ \eta(t)$ и конечные условия, полученные из (16), имеют вид:

$$-\frac{d\varphi(t)}{dt}=\lambda A\varphi_1(t)v(t)-\frac{B^2}{\beta}K(t)\varphi(t)+A\varphi(t)-\lambda\frac{B^2}{\beta}K(t)\varphi_1(t)v(t)-$$
$$-\lambda^2\frac{B^2}{2\beta}K_2(t)\varphi_1(t)v^2(t)-2\gamma_1Qg-\lambda\frac{B^2}{2\beta}K_2(t)\varphi(t)v(t)+\lambda\varphi_1(t)+\lambda\varphi_1(t)G(t,t),$$
$$\varphi(T)=-2\gamma_2Fg; \qquad (20)$$
$$-\lambda\frac{d\varphi_1(t)}{dt}=-\lambda\frac{B^2}{\beta}K_1(t)\varphi(t)v(t)-\lambda^2\frac{B^2}{\beta}K_1(t)\varphi_1(t)v^2(t)+\lambda\varphi(t)-$$
$$-\lambda\frac{B^2}{2\beta}K_2(t)\varphi(t)+\lambda^2\varphi_1(t)v(t)-\lambda^2\frac{B^2}{2\beta}K_2(t)\varphi_1(t)v(t),\quad \varphi_1(T)=0; \qquad (21)$$
$$-\frac{d\eta(t)}{dt}=\gamma_1Qg^2-\lambda^2\frac{B^2}{4\beta}\varphi_1^2(t)v^2(t)-\frac{B^2}{4\beta}\varphi^2(t)-\lambda\frac{B^2}{2\beta}\varphi(t)\varphi_1(t)v(t),\quad \eta(T)=\gamma_2Fg^2. \qquad (22)$$

Таким образом, получены новые вспомогательные уравнения типа Риккати в задаче 1 при $g=0$ и в задаче 2 при $g\neq 0$ и построены соответствующие новые алгоритмы управления. Интегральное слагаемое входит только в уравнение (1) модели управляемого процесса.

В работе [6] в выводе соответствующих уравнений помогли добавки в форму решения $S(t,x)$ уравнения Беллмана и в критерий J. В данной работе показано, что уравнения Риккати можно получить без добавок в традиционный критерий J (2).

Требуется дальнейшая работа в этом направлении.

4. Пример 1, $g = 0$

Рассмотрим пример 1. В задаче 1 в расчетах использована схема Эйлера первого порядка [7]. В программе расчетов для статьи [8], где в критерий качества также были введены интегральные слагаемые, взяли равными нулю параметры $Q1 = 0;\ F1 = 0$. То есть по этой программе расчеты выполнены для критерия (2), вспомогательной системы Риккати (8) – (10), управления (13). На рис. 2 – 4 приведены графики функций $K(t),\ K_1(t),\ K_2(t)$ (8) – (10), на рис. 5, 6 графики управления $u^0(t)$ (13) и соответствующего состояния процесса $x^0(t)$ (1) с начальным условием $x_0 = 30$.

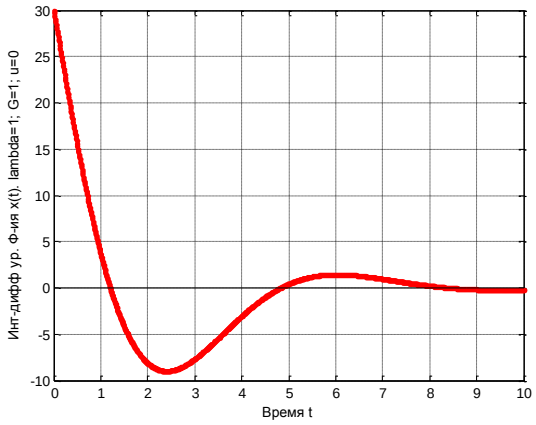


Рисунок 1 – Состояние $x(t)$ при $u(t) \equiv 0$.

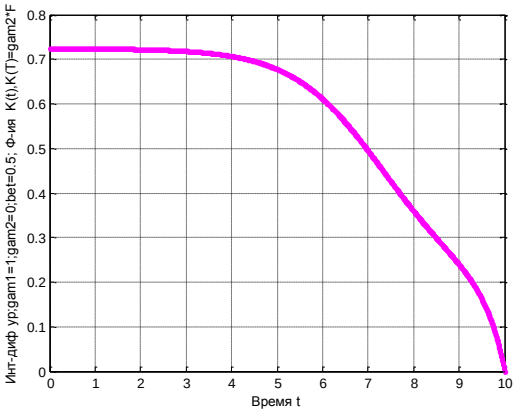


Рисунок 2 – Функция Риккати $K(t)$.

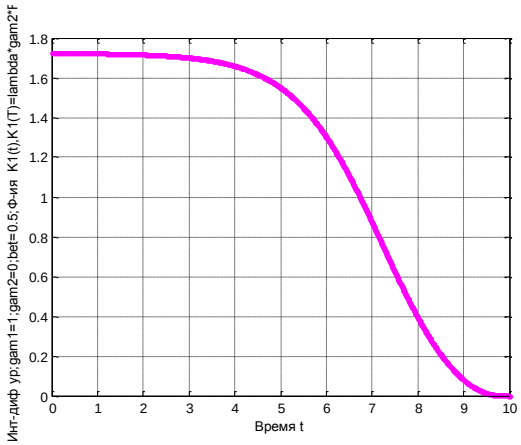


Рисунок 3 – Функция Риккати $K_1(t)$.

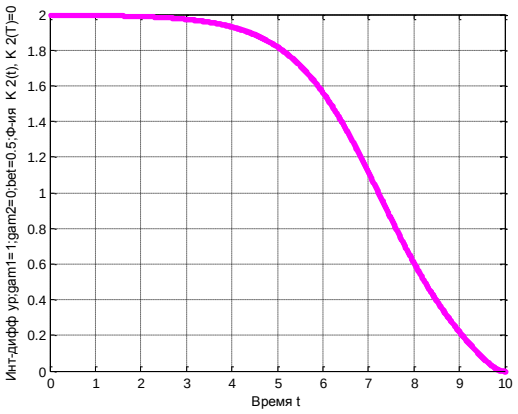


Рисунок 4 – Функция Риккати $K_2(t)$.

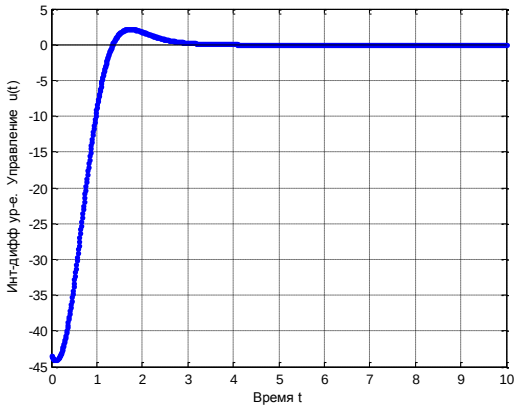


Рисунок 5 –Управление $u^0(t);\ \lambda = 1;\ G = 1$.

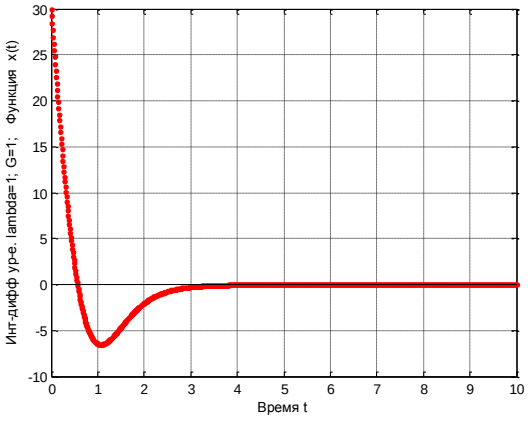


Рисунок 6 – Состояние $x(t)$ при $u^0(t)$.

Получено методом динамического программирования в задаче 1: критерий качества (2) равен $J^0 = 637.8569$, управление в начальной точке $u^0(0) = -43.4678$. Рисунки 1 и 6 показывают, что управление $u^0(t)$ интенсивно переводит состояние процесса $x(t)$ к нулю по сравнению с нулевым управлением.

Заключение

Построены новые алгоритмы управления с обратной связью процессом с последействием, описываемым одномерным интегро-дифференциальным уравнением типа Вольтерра. Компьютерное моделирование показало работоспособность предложенных алгоритмов управления.

Литература

1. Быков Я.В. О некоторых задачах теории интегро-дифференциальных уравнений. – Фрунзе: Киргосуниверситет. Типография №1 Главиздата Министерства культуры Киргизсой ССР, 1957. – 328 с.
2. Егоров А.И. Об асимптотическом поведении решений системы интегро-дифференциальных уравнений типа Вольтерра. Дисс… канд. физ.-мат. наук. Фрунзе, Киргосуниверситет, 1955.
3. Алексеев В.М., Тихомиров В.М., Фомин С.В. Оптимальное управление. – М.: Наука, 1979. 432 с.
4. Люстерник Л.А., Соболев С.Л. Элементы функционального анализа. – М.: Наука, 1965.
5. Фельдбаум А.А., Бутковский А.Г. Методы теории автоматического управления. – М.: Наука, 1971. 744 с.
6. Ройтенберг Я.Н. Автоматическое управление. – М.: Наука, 1978. 552 с.
7. Турчак Л.И. Основы численных методов. – М.: Наука, 1987. – 320 с.
8. Самохвалова Т.П., Керимбеков А.К., Таирова О.К. Интегро-дифференциальное уравнение в модели управляемого процесса // Проблемы автоматики и управления. – 2019. № 1 (36). – С. 77 – 83.

УДК 621.3.019.3 10.5281/zenodo.3904087

Бакасова А.Б., Сатаркулов К., Ниязова Г.Н.
Институт машиноведения и автоматики НАН КР, г. Бишкек
bakasovaaina@mail.ru, gulmira-n.86@mail.ru
Сатаркулов Т.К
Кыгрызский государственный технический университет им. И.Раззакова
satarkulov46k@mail.ru

О МЕТОДЕ АНАЛИЗА НАДЕЖНОСТИ И ДИАГНОСТИКИ СОСТОЯНИЙ МИКРОГЭС ДЛЯ АВТОНОМНОГО ЭЛЕКТРОСНАБЖЕНИЯ

В статье дан краткий обзор микроГЭС с центробежным регулятором и маховиком с автоматически регулируемой массой и моментом инерции (АРМИ). Описан метод анализа надежности и диагностики состояний. МикроГЭС представлена в виде некоторой физической системы S с протекающим в ней случайным процессом Марковского типа с дискретными состояниями и непрерывным временем. Предполагается, что все переходы системы S из одного состояния в другое происходят под действием простейших потоков отказа λ и восстановления μ с заданными интенсивностями. Составлен размеченный граф состояний системы и построена математическая модель рассматриваемого процесса, которая дает возможность найти все вероятности состояний в виде функции времени, т.е. произвести диагностику состояний. Полученная математическая модель позволяет оценить работу микроГЭС с учетом всех условий, от которых зависит ее функциональность, оценить потери, вызванные различными причинами.

Ключевые слова: микроГЭС, маховик, стабилизация частоты вращения, гидротурбина, регулятор Уатта, математическая модель, Марковский процесс, размеченный граф, анализ надежности, диагностика состояний.

Введение. Электроснабжение населенных пунктов, находящихся в зоне децентрализованного электроснабжения, осуществляется в основном от дизельных электростанций. Удаленность и малочисленность таких населенных пунктов, а также низкая потребляемая ими мощность влияют на снижение рентабельности энергетических установок. Поэтому ветровые электростанции [1, 2] и микроГЭС малой мощности [3–6] для таких населенных пунктов все чаще рассматриваются как альтернативные источники энергии. Потенциальными потребителями малых и особенно микроГЭС могут стать удаленные поселки, геологические станции, метеостанции, небольшие фермерские хозяйства, летние пастбища, туристические базы и т.д.

В работах [7–10] приведены конструкции, математические модели и подробные описания принципов работы различных вариантов мобильных микроГЭС. На рис. 1 представлена схема микроГЭС, математическая модель которой описывается системой (1) [9, 10].

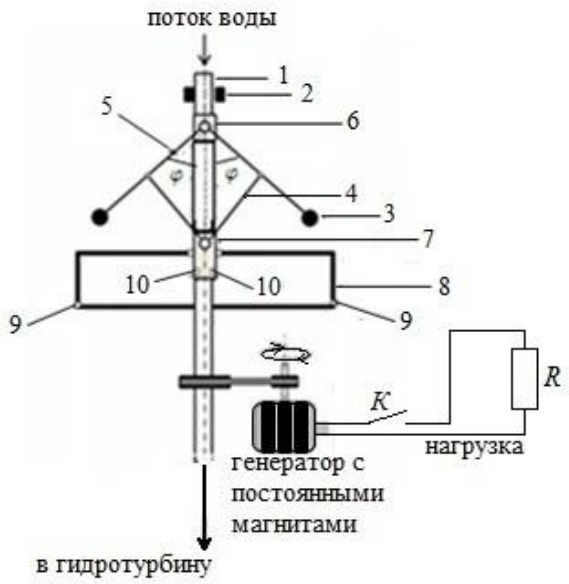


Рисунок 1 – Схема микроГЭС с центробежным регулятором и с АРМИ маховиком: 1 – напорный трубопровод, 2 – подшипник; 3 – металлические грузики; 4 – стержни, шарнирно прикрепленные к подвижной муфте, 5 – стержни, шарнирно прикрепленные к неподвижной втулке 6; 7 – подвижная муфта, надетая на напорный трубопровод; 8 – маховик с перегородкой, разделяющей полость на две части; 9 – сквозные отверстиями для выброса воды из маховика; 10 – сквозные отверстия для поступления воды в полость маховика.

Стабилизация частоты микроГЭС осуществляется маховиком с автоматически регулируемыми массой, моментом инерции (АРМИ маховик) и потоками воды, которые подаются на вход гидротурбин. При заполнении или удалении воды из полостей изменяется масса маховика и его момент инерции. При номинальной скорости вращения гидротурбины $\omega_{ном}$ отверстия 10 закрыты муфтой 7, а полость маховика не заполнена водой. Подробное описание установки и принцип ее работы представлены в [9, 11].

Задачей в данной статье является получение дифференциального уравнения для определения вероятностей состояний микроГЭС для автономного электроснабжения (рис. 1), чтобы в дальнейшем провести анализ надежности и диагностику состояний микроГЭС с центробежным регулятором и с АРМИ маховиком.

Математическая модель системы (дифференциальное уравнение для вероятностей состояний)

МикроГЭС (рис. 1) представим в виде некоторой физической системы S, которая с течением времени меняет свое состояние, переходя из одного в другое случайным образом, т.е. в рассматриваемой системе протекает случайный процесс. Предположим, что вероятностное состояние системы S в будущем для любого момента времени t (при $t > t_0$) зависит только от ее состояния S_i в настоящий момент ($t = t_0$) и не зависит от того, когда и как система пришла в это состояние (при $t < t_0$). Такой случайный процесс носит название Марковского [12, 13].

В нашем случае *Марковский случайный процесс* (МСП) является процессом с дискретными состояниями и непрерывным временем, так как состояния микроГЭС можно заранее перечислить (S_0 – работоспособное, S_1 – аварийный останов из-за повреждения основных элементов микроГЭС или напорного водопровода, S_2 – останов из-за повреждения питающей линии электропередач, S_3 – плановый ремонт) (рис. 2). При этом будем считать, что переход микроГЭС из состояния в состояние происходит «скачком», практически мгновенно.

Для анализа рассматриваемого случайного процесса воспользуемся графом состояний (рис. 2).

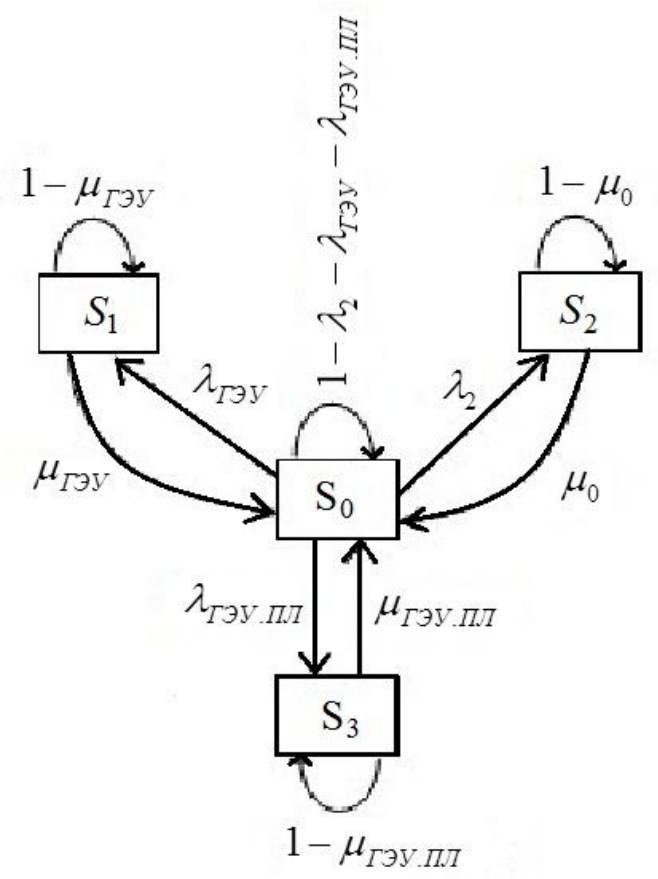


Рисунок 2 – Граф состояний микроГЭС

На рис. 2 λ_2 – интенсивность потока событий (отказов), связанных с повреждением питающей линии электропередач, приводящих к останову микроГЭС; $\lambda_{ГЭУ}$ – интенсивность потока событий, связанных с повреждением основных элементов микроГЭС или напорного водопровода, приводящих к аварийному останову микроГЭС; $\lambda_{ГЭУ.ПЛ}$ – интенсивность потока событий, связанных с плановым ремонтом микроГЭС; μ_0 – интенсивность восстановления линии; $\mu_{ГЭУ}$ – интенсивность восстановления основных элементов микроГЭС или напорного водопровода, $\mu_{ГЭУ.ПЛ}$ – интенсивность восстановления после планового ремонта.

Имея размеченный граф, можно построить математическую модель

рассматриваемого процесса и решить его, найдя все вероятности состояний $p_i(t)$ как функции времени.

Рассмотрим вероятность $p_0(t)$, являющейся вероятностью состояния микроГЭС S_0 в момент *t*. Найдем эту же вероятность по истечении времени Δt. То есть вероятность $p_0(t+\Delta t)$ того, что в момент $t+\Delta t$ система будет в состоянии S_0.

Такое возможно в следующих случаях:

1) в момент *t* система уже находилась в состоянии S_0, и за время Δt она не вышла из него;
2) в момент *t* система была в состоянии S_1, а за время Δt перешла из него в S_0;
3) в момент *t* система была в состоянии S_2, а за время Δt перешла из него в S_0;
4) в момент *t* система была в состоянии S_3, а за время Δt перешла из него в S_0.

Следуя [13] составим уравнения состояний системы: S_0, S_1, S_2, S_3. Для этого определим вероятности перечисленных вариантов.

В первом случае величина $p_0(t)$ равна вероятности того, что в момент *t* система была в состоянии S_0. Эту вероятность умножим на вероятность того, что система, находившаяся в момент *t* в состоянии S_0, за время Δt не перейдет из него ни в S_1, ни в S_2, ни в S_3. Суммарный поток событий, выводящий систему из состояния S_0, равен $\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ}$. Тогда вероятность того, что за время Δt система выйдет из состояния S_0 равна $\Delta t\cdot(\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ})$, а вероятность того, что не выйдет: $1-\Delta t\cdot(\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ})$. Следовательно, вероятность первого варианта равна $p_0(t)\cdot\left[1-\Delta t\cdot(\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ})\right]$.

Во втором случае в момент *t* система находится в состоянии S_1 и за время Δt перейдет из него в состояние S_0, т.е. она равна $p_1(t)\cdot\mu_{ГЭУ}\cdot\Delta t$.

Аналогично находим вероятность для третьего и четвертого случаев, которые соответственно равны: $p_2(t)\cdot\mu_0\cdot\Delta t$ и $p_3(t)\cdot\mu_{ГЭУ.ПЛ}\cdot\Delta t$.

Суммируя вероятности всех вариантов (по правилу сложения вероятностей) получим:

$$\begin{aligned} p_0(t+\Delta t) = {} & p_0(t)\cdot[1-\Delta t\cdot(\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ})]+ \\ & + p_1(t)\cdot\mu_{ГЭУ}\cdot\Delta t + p_2(t)\cdot\mu_0\cdot\Delta t + p_3(t)\cdot\mu_{ГЭУ.ПЛ}\cdot\Delta t. \end{aligned} \tag{3}$$

Раскрыв в выражении (3) квадратные скобки, перенеся $p_0(t)$ в левую часть, разделив обе части на Δt и устремив его к нулю, получим дифференциальное уравнение для $p_0(t)$:

$$\frac{dp_0(t)}{dt} = \mu_{ГЭУ.ПЛ}\cdot p_3(t) + \mu_0\cdot p_2(t) + \mu_{ГЭУ}\cdot p_1(t) - p_0(t)\cdot(\lambda_2+\lambda_{ГЭУ}+\lambda_{ГЭУ.ПЛ}) \tag{4}$$

Рассуждая аналогично для всех остальных состояний, получим еще три дифференциальных уравнения. В итоге получим систему дифференциальных уравнений для вероятностей состояний:

$$\left.\begin{aligned}
&\frac{dp_0(t)}{dt} = \mu_{ГЭУ.ПЛ} \cdot p_3(t) + \mu_0 \cdot p_2(t) + \mu_{ГЭУ} \cdot p_1(t) - p_0(t) \cdot (\lambda_2 + \lambda_{ГЭУ} + \lambda_{ГЭУ.ПЛ}),\\
&\frac{dp_1(t)}{dt} = \lambda_{ГЭУ} \cdot p_0(t) - \mu_{ГЭУ} \cdot p_1(t),\\
&\frac{dp_2(t)}{dt} = \lambda_2 \cdot p_0(t) - \mu_0 \cdot p_2(t),\\
&\frac{dp_3(t)}{dt} = \lambda_{ГЭУ.ПЛ} \cdot p_0(t) - \mu_{ГЭУ.ПЛ} \cdot p_3(t).
\end{aligned}\right\} \quad (5)$$

При этом, следует отметить, что

$$\sum_{i=1}^{4} p_i(t) = 1. \quad (6)$$

Чтобы решить уравнения (5), необходимо задать начальные условия. Очевидно, что в начальный момент $t = 0$, когда все элементы системы исправны, $p_0(0) = 1$, $p_1(0) = 0$, $p_2(0) = p_3(0) = 0$.

В заключении отметим, что система уравнений (5), также называемая уравнением Колмогорова, дает возможность найти все вероятности состояний как функции времени, т.е. произвести диагностику состояний. Кроме того, полученная математическая модель позволяет оценить работу микроГЭС с учетом всех условий, от которых зависит ее функциональность, оценить потери, вызванные различными причинами.

Литература

1. Бобров А.В., Тремясов В.А. Анализ надежности ветроустановок для автономного электроснабжения // Электроэнергия: от получения и распределения до эффективного использования: материалы всероссийской научно-технической конференции, Томск: изд-во ТПУ, – 2008. – С. 112-113.
2. Тремясов В.А., Надежность электроснабжения: учеб. пособие.; – Красноярск: ИПЦ КГТУ, 2006. – 163 с.
3. Кушнир В.Г., Кошкин И.В., Глушко Д.В. Обоснование установки мини-ГЭС для электроснабжения системы освещения моста // Электротехнические и информационные комплексы и системы. – 2018. – № 1. – Т. 14. – С. 13-18.
4. Крылов А.П., Бакштанин А.М. Новые концепции в развитии микрогидроэнергетики. Гидравлика в напорных водоводах микроГЭС // Строительство и архитектура. – 2017. – № 5. – С. 8-14.
5. Спирин Е.А., Никитин А.А., Головин М.П., Карпенко В.В. О выборе типа

микроГЭС и ее оптимальной мощности в зависимости от гидрологических параметров // Всероссийская конференция «Актуальные проблемы машиностроения». – 2014. – С. 543–547.

6. Лукутин Б.В., Обухов С.Г., Шандарова Е.Б. Автономное электроснабжение от микрогидроэлектростанций. – Томск: STT, 2001. – 120 с.
7. Сатаркулов К., Бакасова А.Б., Иманакунова Ж.С., Ниязова Г.Н. Способ стабилизации частоты автономной микроГЭС // Проблемы автоматики и управления. –2014. – № 1 (26). – С. 20-23.
8. Ниязова Г.Н., Сатаркулов К., Кыдырмаева З.С., Яблочников А.М. Разработка компьютерной модели системы стабилизации и управления частотой вращения турбины микроГЭС нового типа // Проблемы автоматики и управления. – 2017. – № 2 (33). – С. 43-51.
9. Бакасова А.Б., Сатаркулов К.А., Ниязова Г.Н., Яблочников А.М., Усубалиева Г.К. «Моделирование микроГЭС малой мощности с маховиком, автоматически регулирующим момент инерции» // Информатика и системы управления. Благовещенск (РФ). 2019. - №1(59). – С. 36-45.
10. Бакасова А.Б., Сатаркулов К.А, Ниязова Г.Н., Яблочников А.М., Усубалиева Г.К. Маховик с автоматически регулируемыми массой и моментом инерции для повышения качества стабилизации частоты микроГЭС // XIII Всероссийское совещание по проблемам управления ВСПУ-2019: Труды [Электронный ресурс] / Под общ. ред. Д.А. Новикова. -М.: ИПУ РАН, 2019. –3286 с.
11. Шаршеналиев Ж.Ш., Сатаркулов К.А., Бакасова А.Б., Ниязова Г.Н. Саморегулирующийся адаптивный маховик для генераторов автономных микроГЭС // Патент Кыргызской Республики № 1743. 2015. 20140113.1 Бюл. №5 (71)(73).
12. Феллер В. Введение в теорию вероятностей и ее приложения, Т. 1. – М.: Мир, 1984. – 527 с.
13. Вентцель Е.С. Исследование операций: задачи, принципы, методология. – 2-е изд., стер. – М.: Наука. Гл. ред. физ. мат. лит., 1988. – 208 с.

УДК 621.318.3 10.5281/zenodo.3904089

И. В.Бочкарев, Д.Н. Садыков, *elmech@mail.ru*
Институт машиноведения и автоматики НАН КР
Кыргызский государственный технический университет им. И. Раззакова

ИССЛЕДОВАНИЕ ПЕРЕХОДНЫХ ПРОЦЕССОВ В ЭЛЕКТРОМЕХАНИЧЕСКОМ ТОРМОЗНОМ УСТРОЙСТВЕ С УЧЕТОМ ДИНАМИКИ ДВИЖЕНИЯ ЯКОРЯ

Проведены исследования переходных процессов, протекающих в электромеханических тормозных устройствах в зависимости от схемы их управления. Рассмотрены законы изменения тока и потокосцепления электромагнита в различных режимах работы. Составлена и проанализирована математическая модель тормозного устройства в виде системы дифференциальных уравнений с учетом особенностей его конструкции и рабочих режимов. На базе математической модели построены имитационные модели тормозного устройства в среде Simulink при различных схемах управления, в том числе с форсировкой размыкания фрикционного узла тормоза. С использованием имитационных моделей проведены исследования переходных процессов в тормозном устройстве в зависимости от изменения его параметров, и даны рекомендации по выбору наиболее целесообразных схем форсировки.
Ключевые слова: тормозное устройство, схема форсировки, переходной процесс, моделирование, математическая модель.

Введение. При проектировании и исследовании различных электроприводов, содержащих электромеханические тормозные устройства с соответствующей системой управления, возникает необходимость в исследовании влияния динамических свойств непосредственно самого тормозного устройства на переходные процессы системы в целом. Основными параметрами тормозных устройств являются величина тормозного момента и параметры быстродействия размыкания и замыкания их фрикционного узла. Данные параметры напрямую влияют не только на динамические параметры электропривода, но и на электрические потери в приводном двигателе и на механические потери в самом тормозном устройстве. Это в свою очередь сказывается на остаточном ресурсе двигателя и тормоза.

Важной частью исследований по данному вопросу является имитационное моделирование систем электропривода и, в частности, тормозных устройств. При этом следует отметить, что, смотря на простоту конструкции электромеханических тормозных устройств, динамические процессы которых схожи с прямоходовым электромагнитом с поступательным движением якоря, математическое описание представляет собой значительную сложность. Поэтому для приведения математической модели к виду, удобному для моделирования и расчетов принимают ряд допущений.

Постановка задачи. Для проведения исследования переходных процессов в электромеханическом тормозном устройстве требуется составить и проанализировать математическую модель тормоза с учетом особенностей его конструкции и рабочих режимов. Это позволит корректно синтезировать имитационную модель, с использованием которой можно будет достоверно исследовать переходные режимы работы ЭМТУ с учетом схемы их управления.

Результаты исследований. В качестве предмета исследования выберем дисковое пружинное нормально замкнутое электромеханическое тормозное устройство (ЭМТУ), широко используемое в тормозных модификациях электродвигателей [1], составляю-

щих основу позиционных электроприводов. Принципиальная конструктивная компоновка такого ЭМТУ показана на рис. 1.

ЭМТУ устанавливается со стороны вентиляционного узла электродвигателя и закрепляется на подшипниковом щите двигателя (на рис. 1 не показан). Он состоит из электромагнита, содержащего магнитопровод 1 и обмотку 2, якоря 3, тормозного диска 4 и тормозного фланца 5. Якоря 3 может передвигаться в осевом направлении по направляющим штифтам 6, неподвижно закрепленных в магнитопроводе. Тормозной диск 4 выполнен в виде упругой мембраны, на которой наклеены тормозные накладки 7, и закреплен на буртике 8 переходной втулки 9, зафиксированной на валу 10 электродвигателя.

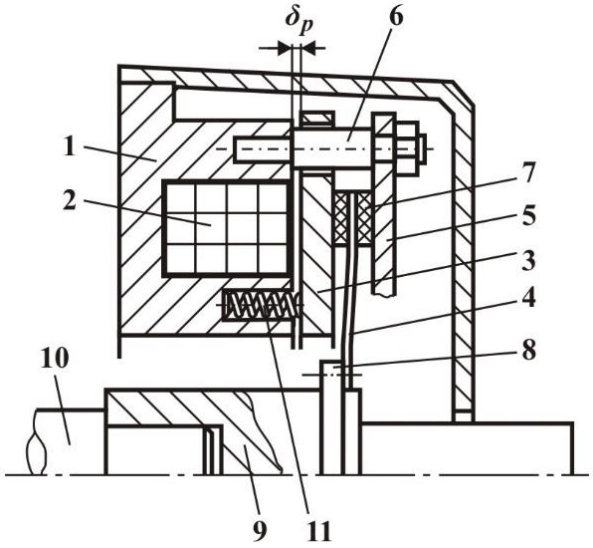


Рисунок 1 – Разрез ЭМТУ по оси вала электрической машины

При обесточенной обмотке 2 тормозной диск 4 зажат между якорем 3 и тормозным фланцем 5 усилием тормозных пружин 11, размещенных в пазах, выполненных в магнитопроводе. За счет этого вал 10 заторможен силами трения. При подаче напряжения на обмотку 2 якорь 3 под воздействием магнитного поля электромагнита притягивается к магнитопроводу 1 и освобождает тем самым тормозной диск 4. Упругая мембрана распрямляется и тормозные колодки 7 перестают контактировать и с якорем, и с тормозным фланцем 5, и на вал перестает действовать тормозной момент.

Так как динамические процессы в ЭМТУ схожи с процессами в электромагнитах, то дифференциальные уравнения, описывающие процессы нарастания тока и движения якоря, буду аналогичны друг другу [2,3].

Дифференциальное уравнение электрического равновесия и описание состояния магнитной цепи электромагнита ЭМТУ будут иметь вид [2,3,4]:

$$u(t) = i(t)R + \frac{d\psi}{dt}; \tag{1}$$

$$\psi = (i, x), \tag{2}$$

где $u(t)$ – напряжение источника питания; i – ток протекающий по катушке электромагнита; R – активное сопротивление катушки электромагнита; ψ – потокосцепление катушки; x – величина перемещения якоря электромагнита.

Изменение тока $i(t)$ имеет сложный характер. Весь рабочий цикл его изменения показан на рис.2.

Весь рабочий цикл можно разбить на четыре режима [1, 5]:

I – режим при отключенном источнике питания, при котором ток в обмотке электромагнита отсутствует и фрикционный узел разомкнут, причем между ним и магнитопроводом имеется зазор $\delta = \delta_p$;

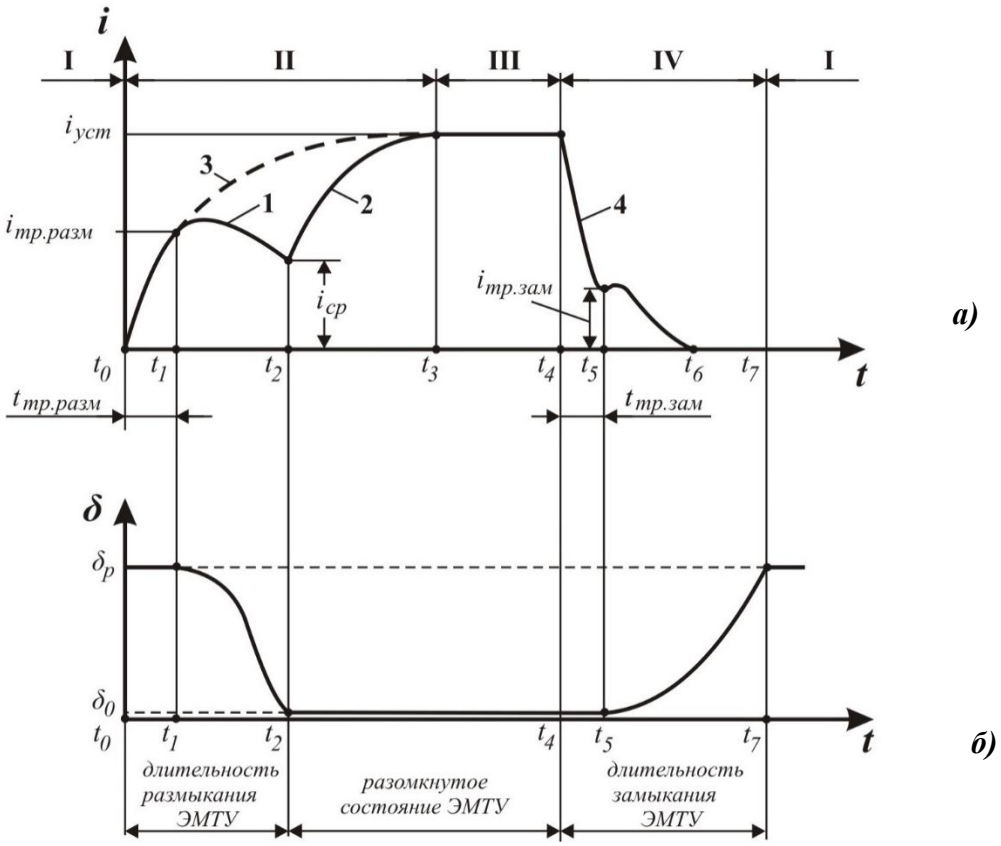


Рисунок 2 – Рабочий цикл электромагнитного привода ЭММ ПМ:
а – зависимость тока в обмотке при питании от источника постоянного напряжения; б – кривая изменения зазора между якорем и магнитопроводом

II – режим подключения обмотки к источнику питания (режим размыкания ЭМТУ)**,** который включает в себя три периода:

- время нарастания тока до величины тока трогания $i_{тр.\ разм}$**,** в течение которого сила притяжения якоря к магнитопроводу постепенно увеличивается и в момент времени t_1 становится равной усилию тормозных пружин, что приводит к исчезновению сил трения в фрикционном узле и его размыканию;

- время движения якоря. В этом периоде работы индуктивность L магнитной системы будет изменяться и уравнение (1) будет иметь следующий вид

$$u(t) = i(t)R + \frac{d\psi}{dt} = i(t)R + L\frac{di}{dt} + i\frac{dL}{dt}. \qquad (3)$$

Таким образом, появляется противодействующая ЭДС $i \cdot dL/dt$, за счет которой ток в обмотке электромагнита уменьшается и в кривой тока появляется характерный провал (линия 1). В момент времени t_2 якорь полностью притягивается к магнитопроводу и упирается в него. В этом состоянии между ними останется остаточный эквивалентный воздушный зазор δ_o, величина которого определяется чистотой обработки сопрягаемых поверхностей и качеством сборки фрикционного узла. Ток при $t_{.} = t_2$ называют током срабатывания $i_{ср}$.

- время нарастания тока до установившейся величины (кривая 2).

Отметим, что если бы якорь не двигался, то нарастание тока в период $t_{.} = t_1 \div t_3$ происходило бы не по кривым 1 и 2, а плавно по пунктирной линии 3;

III – режим разомкнутого состояния фрикционного узла ($t_{.} = t_3 \div t_4$), при котором ток равен установившемуся значению $i_{уст}$;

IV – режим отключения обмотки от источника питания (режим замыкания ЭМТУ) Этот режим включает в себя три периода:

- время уменьшения тока до величины тока трогания $i_{тр.зам}$, в течение которого сила притяжения якоря к магнитопроводу постепенно снижается и в момент времени $t_{.} = t_5$ становится равной усилию тормозных пружин;

- время движения якоря от магнитопровода к тормозному диску, в течение которого опять наводится ЭДС самоиндукции и ток за счет этого может увеличиться относительно $i_{тр.зам}$. В момент времени t_7 якорь полностью прижимает тормозной диск 4 к тормозному фланцу 5 (рис. 1);

- время затухания тока до нуля, причем период $t_5 \div t_6$ может быть как меньше, так и больше периода $t_5 \div t_7$.

Изменение потокосцепления ψ также имеет сложный характер, поскольку в соответствие с (2) ψ зависит и от тока i обмотки, и от координаты перемещения якоря (рис. 3).

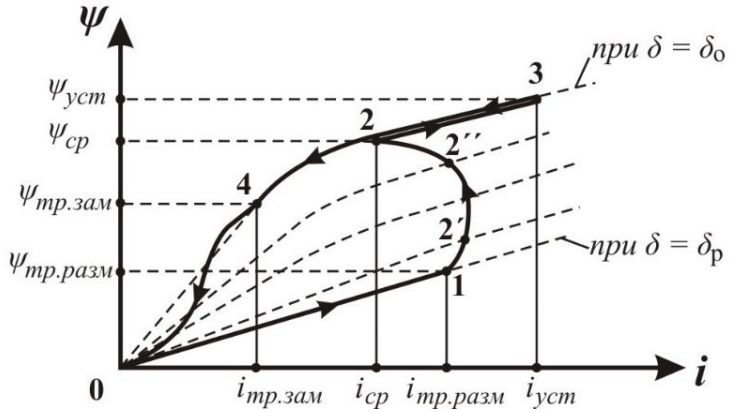


Рисунок 3 – Зависимость потокосцепления $\psi(i, x)$ от изменения тока в катушке и положения якоря относительно магнитопровода

Точка 0 соответствует состоянию, при котором питание отключено, ЭМТУ замкнут и зазор δ между якорем и магнитопроводом равен номинальному значению $\delta = \delta_р$. При подаче напряжения на обмотку ток начинает возрастать (рис. 2). Пропорционально току начинает увеличивается потокосцепление ψ по прямой линии до точки 1, в которой ток становится равным току трогания $i_{тр.разм}$, при котором якорь начнет движение. За счет этого зазор в магнитной цепи начинает уменьшаться и крутизна линий $\psi = f(i)$ будет увеличиваться (на рис. 3 они показаны пунктирными кривыми между линиями при $\delta = \delta_р$ и $\delta = \delta_о$). Поскольку при движении якоря ток будет снижаться (см. рис. 2, кривая 1), то величина потокосцепления ψ будет изменяться нелинейно по точкам 2', 2'' и т.д., которым соответствует значение перемещения якоря $x(t) \in (\delta_0 : \delta_р)$, вплоть до момента полного размыкания фрикционного узла ($i = i_{ср}$, точка 2) и до установившегося режима ($i = i_{уст}$, точка 3).

После отключения питания ток начинает снижаться до тока трогания $i_{тр.зам}$ (точка 4), при котором якорь начнет движение и опять замкнет фрикционный узел. В этом режиме работы ЭМТУ величина потокосцепления ψ между точками 4 и 0 опять будет изменяться нелинейно.

При составлении математической модели ЭМТУ принимаем следующие допущения: 1) при передвижении якоря не учитывается сила трения по направляющим штифтам; 2) изменение индуктивности электромагнита при перемещении якоря $L(x)$ происходит по линейному закону, описываемому следующим уравнением:

$$L(x) = \frac{L_p \delta_p}{\delta_p - x} = \frac{1}{a - bx}, \tag{4}$$

где L_p – индуктивность при замкнутом ЭМТУ при $\delta = \delta_p$; x – величина перемещения якоря.

Для удобства построения имитационной модели представим выражение (4) в виде

$$L(x) = \frac{1}{a - bx}, \tag{5}$$

где $a = 1/L_p$, $b = 1/(L_p \cdot \delta_p)$ – коэффициенты полинома первого порядка.

Для точного определения зависимости $L(x)$ необходим расчет магнитного поля с учетом реальной геометрии магнитопровода ЭМТУ.

Система уравнений, описывающая законы изменения тока в обмотке ЭМТУ и движение якоря электромагнита [3, 6], можно представить в следующем виде

$$\left.\begin{aligned} U &= Ri + L(x)\frac{di}{dt} + i\frac{dL(x)}{dx}\cdot v; \\ m_я \frac{dv}{dt} &= \frac{1}{2} i^2 \frac{dL(x)}{dx} + m_я g - (F_0 + cx); \\ \frac{dx}{dt} &= v, \end{aligned}\right\} \tag{6}$$

где U – амплитуда питающего напряжения ЭМТУ; $L(x)$ – изменение индуктивности электрической цепи электромагнита в зависимости от положения якоря; $m_я$ – масса якоря электромагнита; v – линейная скорость движения якоря электромагнита; F_0 и c – сила начального сжатия и жесткость тормозной пружины.

Изменение тормозного момента ЭМТУ описывается уравнением:

$$M_т = z \cdot k_{тр} \cdot R_{ср}(F_{пр} - F_{эм}), \tag{7}$$

где z – число пар поверхностей трения; $k_{тр}$ – коэффициент трения; $R_{ср}$ – средний радиус сил трения в фрикционном узле; $F_{пр}$ – осевое усилие тормозных пружин; $F_{эм}$ – электромагнитное тяговое усилие растормаживающего электромагнита.

Тяговое усилие электромагнита определяется согласно уравнению

$$F_{эм}(x) = \frac{1}{2} i^2 \frac{dL(x)}{dx}. \tag{8}$$

Тогда учитывая уравнения (4) ÷ (8) и приводя систему (6) к нормальной форме Коши, получаем систему, описывающую электромагнитные и механические процессы в пружинном электромагнитном тормозном устройстве:

$$\left.\begin{aligned} \frac{di}{dt} &= (a - bx)\cdot\left[U - Ri - \frac{b}{(a - bx)^2} iv\right]; \\ \frac{dv}{dt} &= \frac{1}{m_я}\left[\frac{1}{2} i^2 \frac{b}{(a - bx)^2} + m_я g - F_0 - cx\right]; \\ \frac{dx}{dt} &= v; \\ M_т &= z \cdot k_{тр} \cdot R_{ср}\left(F_{пр} - \frac{1}{2} i^2 \frac{b}{(a - bx)^2}\right). \end{aligned}\right\} \tag{9}$$

где $v = dx/dt$ – скорость движения якоря.

Система (9) представлена в виде, необходимом для построения имитационной модели (рис. 4) в среде Simulink [7].

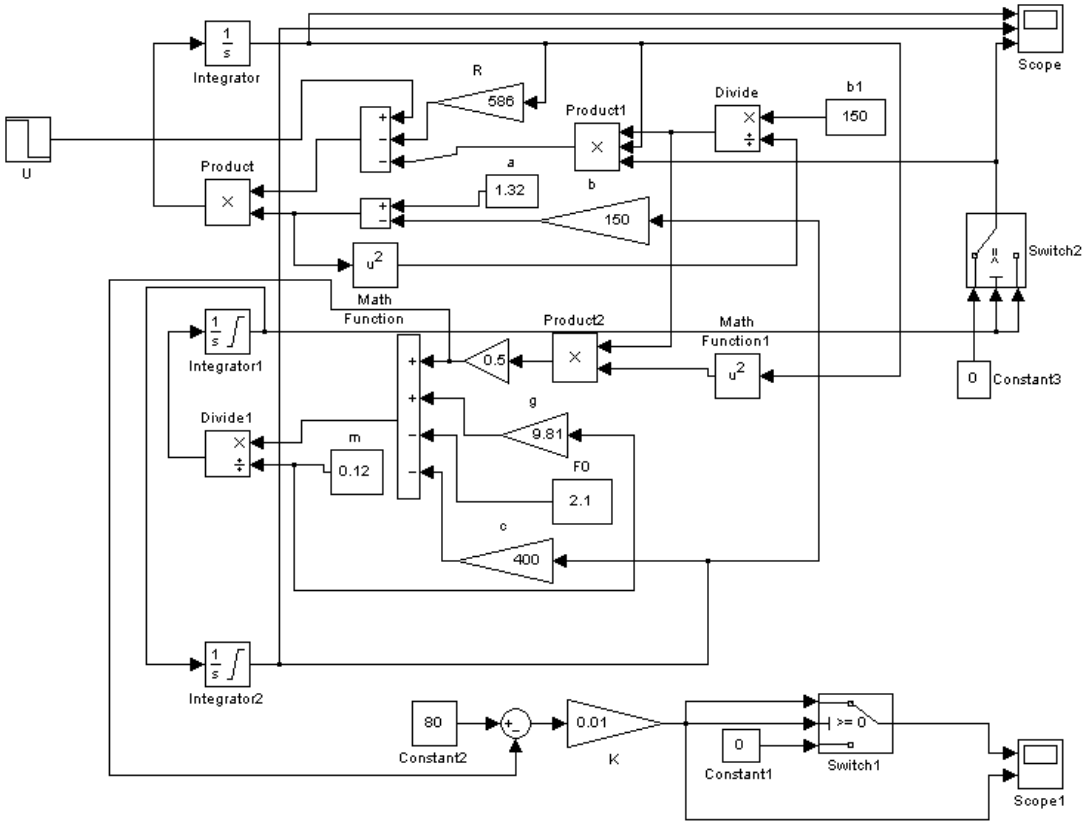


Рисунок 4 – Имитационная модель пружинного электромеханического тормозного устройства

Исследование проведены для трех случаев включения обмотки электромагнита – без форсировки, с форсировкой растормаживания по схеме включения ЭМТУ согласно рис. 7 и с форсировкой растормаживания, при котором повышенное напряжение питания на обмотку ЭМТУ подается вплоть до полного размыкания фрикционного узла [8].

Исходные данные, применяемые для построения имитационной модели ЭМТУ, представлены в таблице 1.

Таблица 1. Исходные данные имитационной модели ЭМТУ

U, В	R, Ом	a	b	c, Н/м	$m_я$, кг	F_0, Н	z	$k_{тр}$	$R_{ср}$, м
100	586	1.32	150	400	0.12	2.1	2	0.4	0,125

Осциллограммы, иллюстрирующие переходные процессы при работе ЭМТУ без форсировки при подаче напряжения на обмотку, представлены на рис. 5. Графики изменения тока и тормозного момента ЭМТУ при включении и последующем отключении питания ЭМТУ представлены на рис. 6. Полное время размыкания фрикционного узла составляет $t_{разм}$ = 0,071 с при величине тока трогания размыкания $i_{тр.разм}$ = 0,16 А. Время замыкания составляет $t_{зам}$ = 0,063 с при величине тока трогания замыкания $i_{тр.зам}$ = 0,158 А. Данные соответствуют номинальному напряжению питания.

Расчетное изменение момента ЭМТУ (рис. 6) представляет собой решение уравнения (7). Но так как при достижении равенства между осевым усилием тормозных

пружин и электромагнитным тяговым усилием растормаживающего электромагнита, момент становится равным нулю и при дальнейшем увеличении электромагнитного тягового усилия, становится отрицательным, то принимается, что фактический тормозной момент (рис. 6) будет определяться только положительными величинами. При отрицательных расчетных значениях момента его фактическое значение равно нулю.

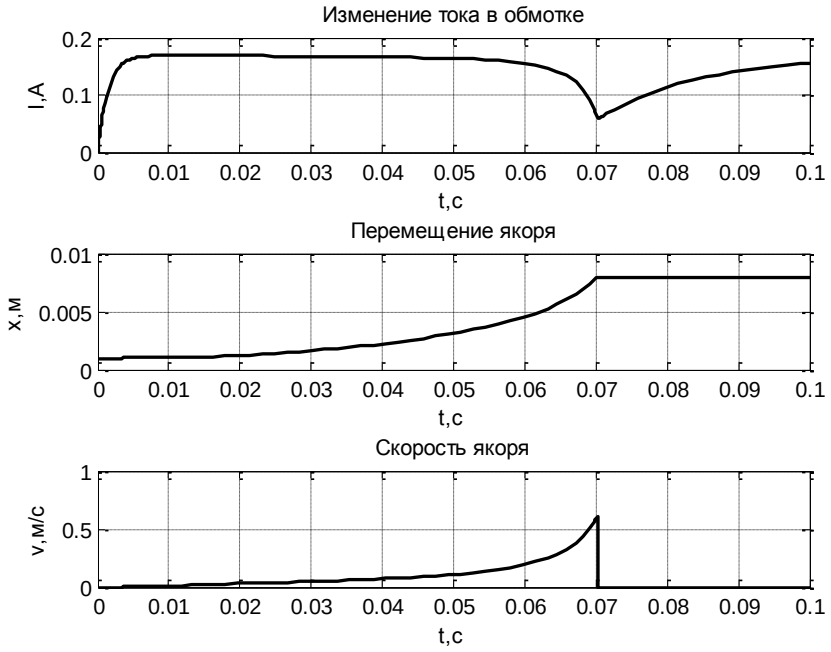


Рисунок 5 – Переходные процессы в ЭМТУ без форсировки

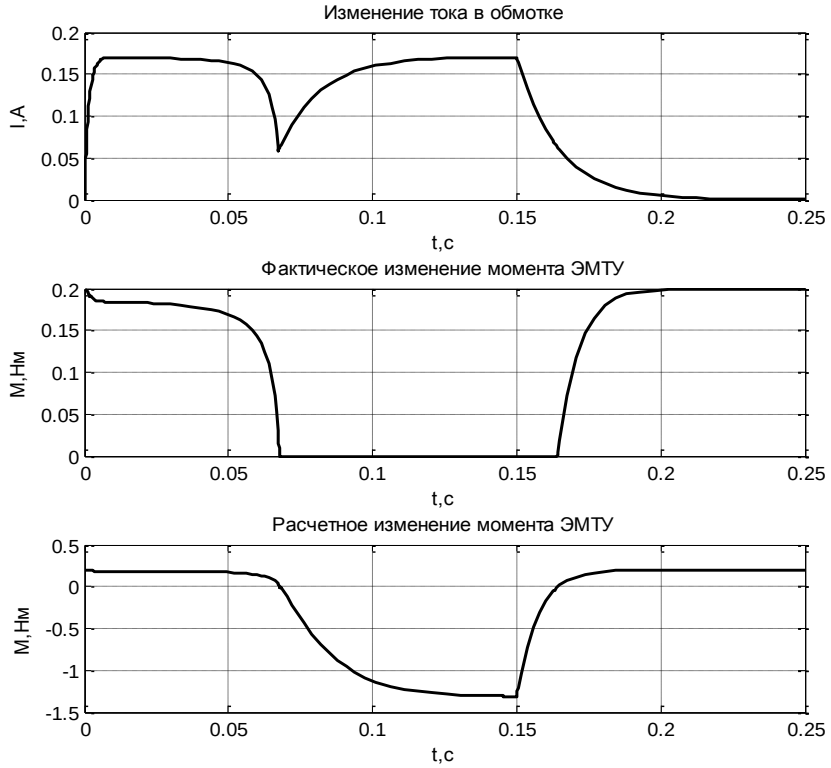
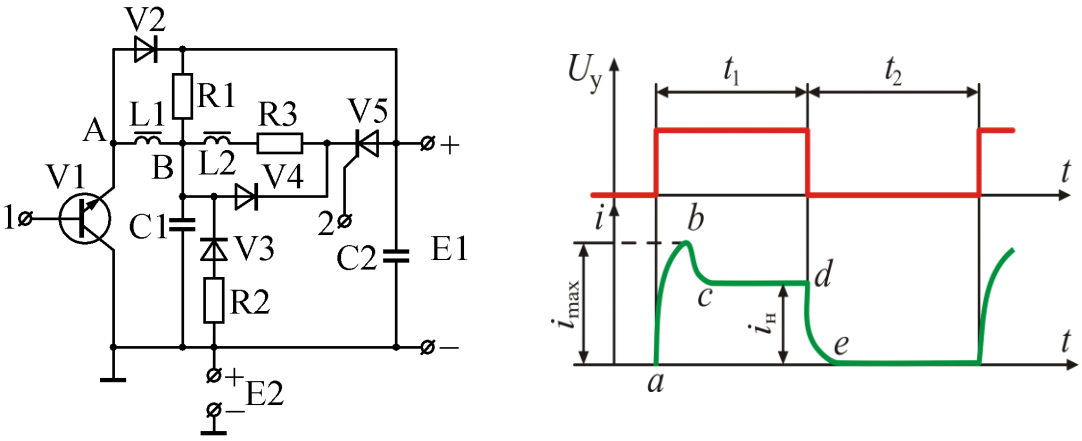


Рисунок 6 – Переходные процессы при включении и последующем отключении питания обмотки ЭМТУ

Результаты моделирования работы ЭМТУ на имитационной модели (рис. 4), приведенные на рис. 5 и 6, схожи с теоретическими зависимостями (рис. 2), что говорит о работоспособности имитационной модели.

Для комплексного улучшения технико-экономических показателей ЭМТУ целесообразно использование схем форсировки размыкания. Существует большое количество различных схем форсировки [1], которые обеспечивают улучшение параметров быстродействия ЭМТУ. В качестве схемы форсировки с использованием емкостного накопителя принимается схема, приведенная на рис. 7.



а) ***б)***

Рисунок – 7. Схема управления с предварительно заряженным конденсатором для тормозного устройства электропривода вязальных машин:
а – схема управления; б – графики изменения управляющего напряжения $U_y(t)$ и тока i(t) в обмотке электромагнита

Подробное описание этой схемы приведено в [9]. В конкретном случае рассмотрим работу схемы в интервале времени от подачи управляющего воздействия на включение питания обмотки ЭМТУ до времени полного размыкания фрикционного узла.

Схема питается от двух источников *E1* и *E2*, разделенных диодом *V4*, причем напряжение *E1* больше напряжения *E2*. В исходном состоянии ключи *V1* и *V5* заперты, а конденсатор *C1* заряжен через резистор *R1* до напряжения источника питания *E1*. До такого же напряжения заряжен и конденсатор *C2*. При поступлении управляющего импульса на вход *1* транзистор открывается, то есть ключ *V1* включается, и конденсатор *C1* разряжается через обмотку электромагнита и переход «коллектор – эмиттер» ключа *V1* (участок *ab* на рис. 7,*б*). При этом ток через обмотку *L1* электромагнита нарастает до значения i_{max}. Крутизна подъема участка *ab* определяется напряжением конденсатора *C1* и постоянной времени цепи его разряда. Как только напряжение на обкладке конденсатора *C1* снизится до напряжения источника *E2*, ток через обмотку электромагнита начнет проходить от источника *E2* через резистор *R2*, диод *V3* и ключ *V1* (участок *cd* на рис.7,*б*) и, соответственно, снижается до номинального значения $i_н$.

Параметры схемы форсировки (рис. 7,*а*) для применения на имитационной модели ЭМТУ следующие: напряжение питания источника E2=U=100 В; напряжение питания источника E1 находится в диапазоне [200:400] вольт и изменяется с шагом 100 В для получения ряда характеристик. Емкость конденсатора C1 постоянна и равна 20 мкФ. Параметры обмотки электромагнита L1 соответствуют исходным данным табл. 1.

Экспериментальные графики переходных процессов в обмотке и фрикционном узле ЭМТУ приведены на рисунках 8 и 9. Ряд характеристик получен при изменении напряжения E1 (т.е. напряжения на конденсаторе C1) в момент разряда на обмотку ЭМТУ.

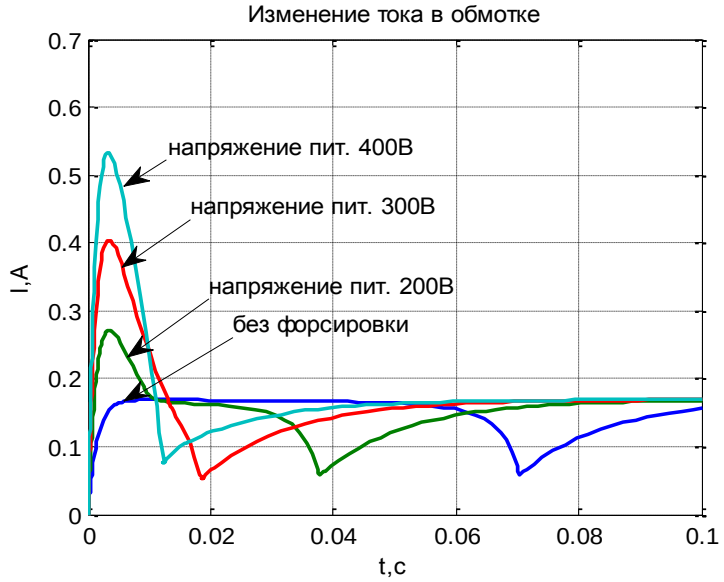


Рисунок 8 – Осциллограммы изменения тока в обмотке ЭМТУ при форсировке с использованием емкостного накопителя

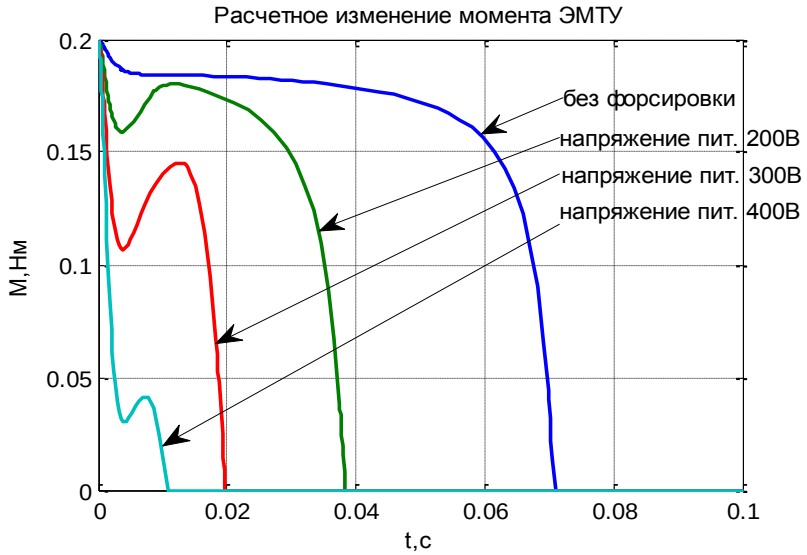
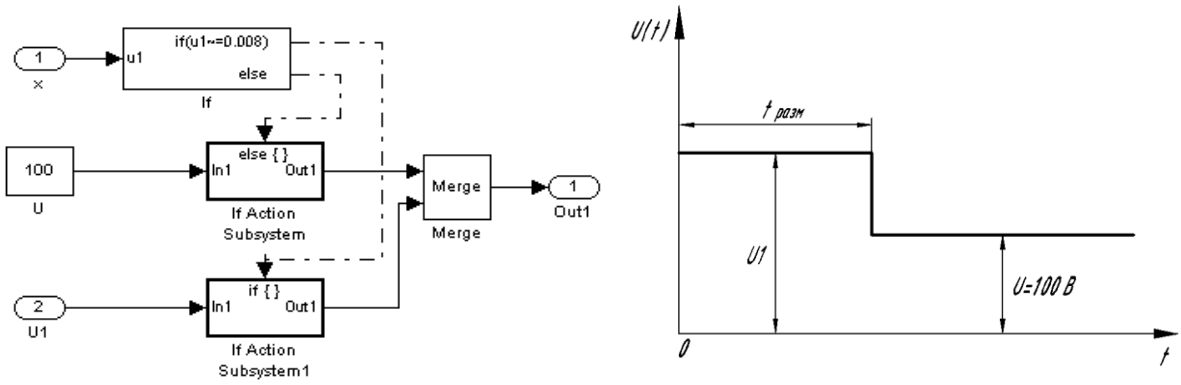


Рисунок 9 – Осциллограммы изменения тормозного момента ЭМТУ при форсировке с использованием емкостного накопителя

При разряжении конденсатора на обмотку ЭМТУ происходит скачок тока с пиком при 0,004 с (рис. 8), причем во всех случаях изменения напряжения на конденсаторе С1 (рис. 7) этот пик тока в обмотке приходится на равное время. Данный скачок приводит к увеличению электромагнитного тягового усилия растормаживающего электромагнита, поэтому на осциллограммах изменения тормозного момента ЭМТУ появляется провал, по времени совпадающий со скачком тока. Следует отметить, что во всех случаях минимальный ток в обмотке ЭМТУ в момент полного размыкания фрикционного узла повышается незначительно и находится в диапазоне [0,065:0,08] ампер.

При форсировке размыкания фрикционного узла ЭМТУ целесообразно подавать повышенное напряжение питания обмотки вплоть до полного размыкания, что можно обеспечить путем использования системы форсировки с двумя источниками питания. Данный способ форсировки реализован с помощью имитационной модели (рис. 10) которая основывается на условном операторе *if* в среде Simulink.



а) ***б)***

Рисунок 10 – Имитационная модель форсировки размыкания фрикционного узла

Рассмотрим работу модели (рис. 10, *а*) на промежутке времени от подачи команды на размыкание фрикционного узла до времени его полного размыкания. Модель образует напряжение питания обмотки ЭМТУ ступенчатого вида (рис. 10, *б*). В модели имеется три входных величины:

- величина перемещения якоря ЭМТУ *x* (рис. 10, *а*). Данный параметр сравнивается с величиной рабочего воздушного зазора, при достижении которого определяется время полного размыкания фрикционного узла ЭМТУ $t_{разм}$ (рис. 10,*б*);

- величина повышенного напряжения питания обмотки ЭМТУ *U*1 (рис. 10, *а*) на промежутке времени $t \in [0{:}t_{разм}]$ (рис. 10,*б*) задается величиной постоянной и при достижении времени $t_{разм}$ скачком становиться равной нулю. Для получения ряда характеристик напряжение *U*1 задается в диапазоне в диапазоне [200:400] вольт и изменяется с шагом 100 В;

- величина номинального напряжения питания ЭМТУ *U* = 100 В (рис. 10, *а*), которая задается как постоянная величина. На промежутке времени $t \in [0{:}t_{разм}]$ равна нулю, при полном размыкании фрикционного узла ЭМТУ, т.е. в установившемся режиме постоянна (рис. 10,*б*).

Экспериментальные графики переходных процессов в обмотке и фрикционном узле ЭМТУ при применении модели (рис. 10, *а*) приведены на рис. 11 и 12.

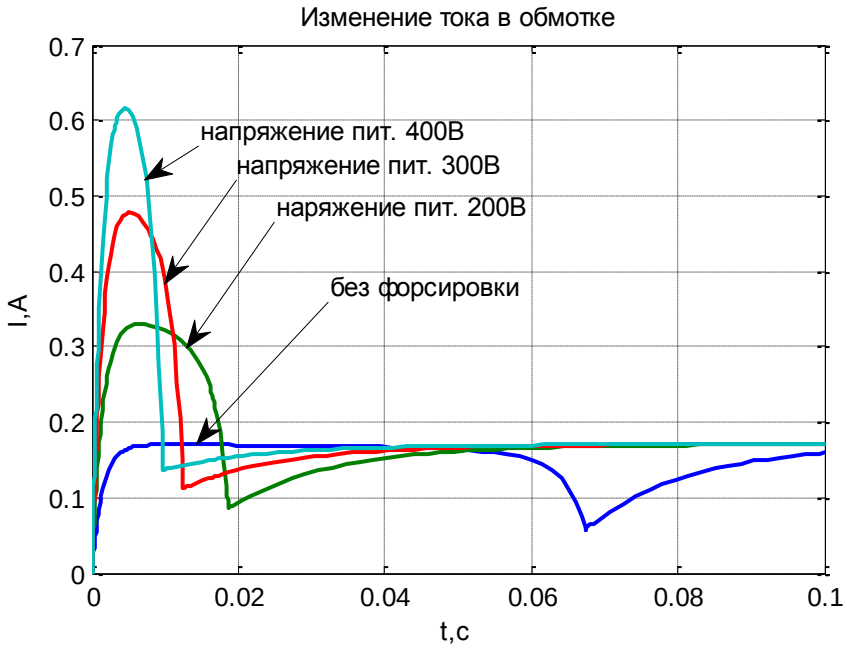


Рисунок. 11– Осциллограммы переходных пропроцессов в обмотке ЭМТУ при подаче напряжения ступенчатого вида

В легенде графика указаны напряжения питания *U*1, которые относятся и к (рис. 11), и к (рис. 12). С использованием модели форсировки (рис. 10,а) значительно уменьшается провал тока в обмотке ЭМТУ (рис. 11) в момент времени полного размыкания фрикционного узла по сравнению с форсировкой с предварительно заряженным конденсатором.

Изменение тормозного момента ЭМТУ (рис. 12) происходит без провалов и за значительно меньшее время размыкания фрикционного узла в сравнении с графиками на рис. 9. Это свидетельствует о том, что применение форсировки с двумя источниками питания позволяет добиться плавного нарастания электромагнитного тягового усилия растормаживающего электромагнита практически с момента подачи питания на обмотку ЭМТУ.

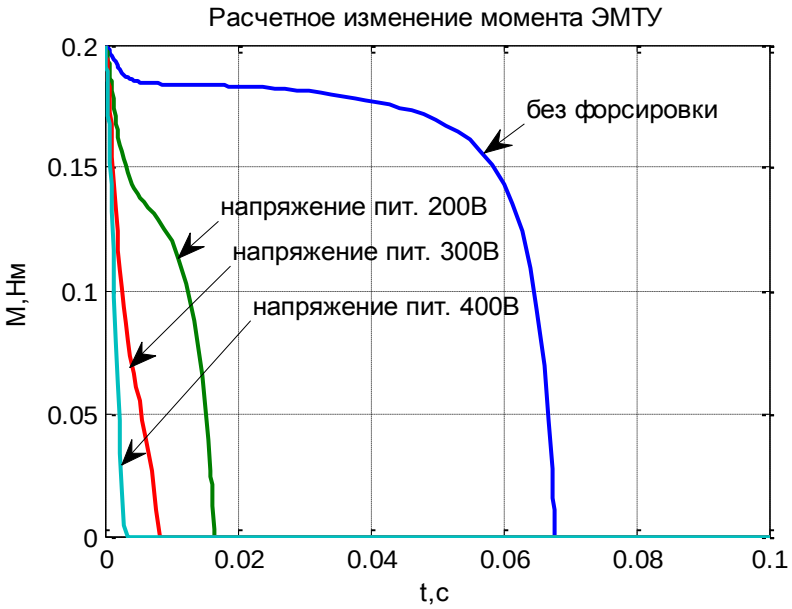


Рисунок 12 – Осциллограммы переходных процессов в фрикционном узле ЭМТУ при подаче напряжения ступенчатого вида

График изменения времени размыкания фрикционного узла ЭМТУ $t_{разм}$ от кратности форсировки $k = E2/U$, приведен на рис. 13.

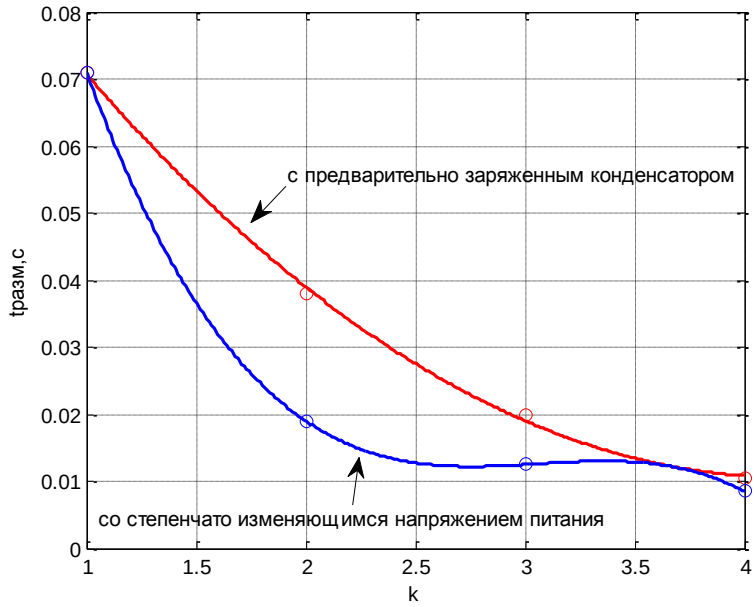


Рисунок 13 – Графики изменения времени размыкания $t_{разм}$ фрикционного узла ЭМТУ в зависимости от кратности отношения максимального напряжения к номинальному

Согласно экспериментальным данным полученным в ходе исследований процессов в ЭМТУ с использованием схем форсирования размыкания фрикционного узла получены интерполированные зависимости времени размыкания в зависимости от кратности отношения максимального напряжения к номинальному (рис. 13). Согласно данным зависимостям целесообразнее применять повышение напряжения вплоть до полного размыкания фрикционного узла ЭМТУ (рис. 10). Также следует отметить, что уменьшение времени полного размыкания фрикционного узла ЭМТУ начиная с 0,02 с в данном случае не целесообразно (рис. 13) для обоих способов форсировки.

Заключение. В результате исследования была показана практическая реализуемость имитационного моделирования пружинного электромеханического тормозного устройства в среде Simulink. Показано, что разработанные имитационные модели позволяют определить численные значения всех основных выходных параметров ЭМТУ. Получены результаты имитационного моделирования ЭМТУ в виде осциллограмм переходных процессов для различных способов управления ЭМТУ – без форсировки и при использовании схем форсированного включения. Оценена эффективность работы схем форсирования размыкания фрикционного узла ЭМТУ для двух случаев: в схеме с использованием емкостного накопителя и с использованием схемы с двумя источниками питания. Получено, что схема форсировки со ступенчатым уменьшением питающего напряжения после размыкания фрикционного узла ЭМТУ более эффективна.

Таким образом, разработанные компьютерные модели позволяют уже на стадии проектирования оценить выходные параметры и динамические характеристики ЭМТУ и выбрать наиболее целесообразную схему его управления.

Литература

1. Бочкарев И.В. Быстродействующие электромеханические тормозные устройства для электродвигателей. – М.: Энергоатомиздат, 2001. – 287 с.
2. Татевосян А.С., Татевосян А.А. и др. Динамика электромагнитов. – Омск: Омский государственный технический университет, 2016. – 148 с.
3. А.С. Татевосян, А.А. Татевосян, Н.В. Захарова. Расчет динамики электромагнита в программе MATLAB // Динамика систем, механизмов и машин, 2016, №2. – С. 174-180.
4. Баранов П.Р., Шараевский А.А. Расчет электромагнитного привода дисковых тормозных устройств асинхронных двигателей с заданным быстродействием // Интернет-журнал «Науковедение». 2013, № 3 (16) [Электронный ресурс]. – М.: Науковедение, 2013 г. – http://nankovedenie.ni/sbomik6/4.pdf.
5. Сливинская А.Г. Электромагниты и постоянные магниты. – М.: Энергия, 1972. – 248 с.
6. Татевосян А.С., Пимонова У.В., Поляков Д.А., Шелковников С.В., Шелковникова Ю.В. Уравнения динамики электромагнита постоянного тока и исследование его динамических характеристик // Современные проблемы науки и образования. – 2015. – № 1 (часть 1).
7. Черных И.В. Моделирование электротехнических устройств в MATLAB, SimPowerSystems и Simulink. – М.: ДМК Пресс; СПб.: Питер, 2008. – 288 с.
8. Бочкарев И.В., Галбаев, Ж.Т. Выбор топологии схемы управления электромагнитными механизмами со встроенным выпрямителем / Проблемы автоматики и управления. – Б.: Илим, 2010. – С.143-148.
9. Лударь А.И., Рабинович Е.Б. Средства автоматики и вычислительной техники для трикотажного оборудования. – М.: Легпромбытиздат, 1989. – 296 с.

УДК 620.953:621.867.1 10.5281/zenodo.3904092

И. В. Бочкарев, А.Р. Сандыбаева, Х.Г. Багиев , elmech@mail.ru
Институт машиноведения и автоматики НАН КР
Кыргызский государственный технический университет им. И. Раззакова

СИСТЕМА УПРАВЛЕНИЯ ЭЛЕКТРОПРИВОДОМ БИОЭНЕРГЕТИЧЕСКОГО КОМПЛЕКСА

Показана целесообразность переработки различных органических отходов с целью получения из них полезных веществ. Рассмотрена структура биоэнергетического комплекса и последовательность технологических операций при его работе. Описан самый первый этап работы комплекса – процедура навозоудаления. Показано, что для реализации этой процедуры используется механический способ удаления навоза посредством системы скребковых навозоуборочных транспортеров, которая состоит из двух транспортеров – горизонтального и наклонного, каждый из которых имеет собственный приводной двигатель и работает независимо друг от друга. Показано, что работа навозоуборочных транспортёров характеризуется целым рядом специфических особенностей, которые необходимо учесть при разработке схемы их управления. Разработана схема частотного управления транспортеров, основу которой составляет программируемый логический контроллер.

Ключевые слова: биоэнергетический комплекс, биореактор, система навозоудаления, скребковый транспортер, схема управления транспортеров, программируемый логический контроллер, преобразователь частоты.

Введение. Переработка и утилизация отходов является серьезной проблемой в большинстве развитых стран, поскольку она связана с серьезными экологическими и экономическими вопросами. Эта проблема касается всех видов отходов – бытовых, производственных, сельскохозяйственных и, в частности, отходов животноводства. Так, крупные фермы, имеющие 10 000 и более животных, оказывают значительное влияние на экосистему и представляют проблемы, которые трудно решить как в краткосрочной, так и в долгосрочной перспективе. В попытке предотвратить экологическую катастрофу, в разных странах законодательно установлены ограничения на количество отходов животноводства, которые могут распространяться в год на единицу поверхности. Например, согласно Регламента № 1069/2009 Европейского парламента и совета, который вступил в силу с 4 марта 2011 г., в Европе введено ограничение, оговаривающее содержание азота в отходах [1], которое фактически ограничивает размер свиноферм. Поэтому вопрос утилизации экскрементов является весьма проблематичным и актуальным.

Конечно, отходы животноводства (навоз и помет) можно использовать в качестве удобрения. Однако эти отходы в свежем виде применять как удобрение нельзя, их необходимо подвергнуть процессу перепревания, чтобы они превратились в перегной, пригодный для удобрения почвы. Этот процесс довольно сложный и занимает длительное время (обычно не менее 12 месяцев). Причем очевидно, что для этого требуется отчуждения больших площадей сельскохозяйственных земель под хранение навоза. Например, в России хранением навоза занято более двух миллионов гектаров земли. Следовательно, этот способ утилизации вызывает большое количество проблем, причем как финансовых, так и экологических, т.к. для перевозки и длительного хранения больших объемов отходов необходимы большие затраты, а их хранение оказывают не только сильное негативное влияние на экологическую среду, но и приводит к накоплению в травах, зерне и источниках воды вредных веществ и может даже привести к вспышке опасных инфекций.

Таким образом, существует потребность в эффективных и экономически.

Использование биоэнергетических установок для переработки органических отходов. Одним из путей решения указанной проблемы является использование отходов животноводства для получения биогаза (метана) и удобрений путем их анаэробной переработки в биоэнергетических установках [2–4]. Такой подход не только позволяет произвести утилизацию отходов с целью обеспечения их повторного использования в народном хозяйстве, но и одновременно решает задачу производства энергии из возобновляемых ресурсов. Благодаря своей постоянной доступности и независимости, в отличии от других возобновляемых источниках, от постоянных изменений внешних условий, например, интенсивности ветра или солнечного света, биоэнергия вносит значительный вклад в дополнение к энергетическому сочетанию возобновляемых источников энергии [5]. В условиях Кыргызстана получаемый таким способом биогаз наиболее целесообразно использовать для отопления зданий, а также для поддержания требуемой температуры протекания самого биогазового процесса [6, 7].

Производство термически пригодных газовых смесей из биоразлагаемых субстратов, т.е. извлечение из них биогаза и преобразования тем самым биомассы в энергию, представляет собой комплексный технический процесс, включающий целый ряд этапов. В целом система биоэнергетического комплекса показана на рис.1.

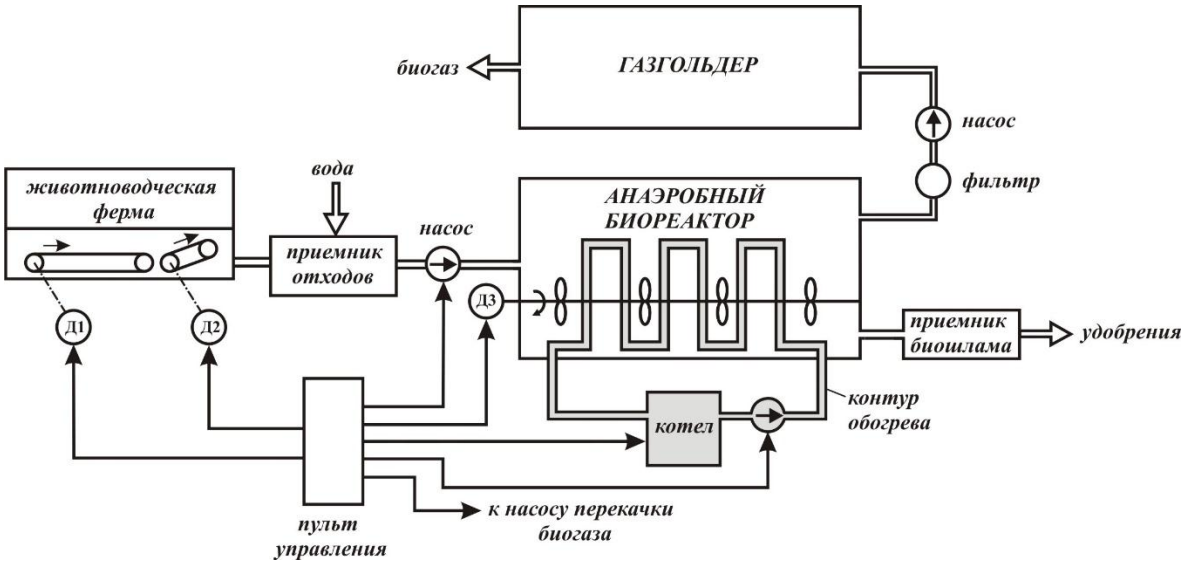


Рисунок 1 – Принципиальный состав системы биоэнергетического комплекса

В состав биоэнергетического комплекса входят следующие основные структурные составляющие:

– животноводческая ферма, оборудованная системой навозоудаления;
– приемник отходов, в котором биомасса подготавливается к переработке;
– биореактор, в которой биомасса подвергается анаэробному брожению;
– газгольдер, представляющий собой резервуар для сбора биогаза;
– приемник биошлама;
– пульт управления.

Последовательность технологических операций при работе биоэнергетического комплекса следующая. На этапе подготовки исходная биомасса собирается и удаляется из животноводческой фермы посредством системы специальных навозоуборочных транспортеров, приводимых в действие электродвигателями Д1 и Д2. Затем она подается в специальную емкость, где смешивается с водой для проведения процесса мокрого сбраживания. Этот процесс проводится непосредственно перед подачей подготовлен-

ного субстрата в биореактор. В нем для получения биогаза создают условия, благоприятные для развития определенных видов бактерий, которые в процессе жизнедеятельности выделяют метан. Кислород для их жизнедеятельности не требуется, поэтому процесс осуществляется анаэробным способом [8]. При этом большое значение имеет не только состав и консистенция сырья [9], но также температура и внутреннее давление.

В сущности, процесс анаэробного сбраживания включает в себя два этапа. На первом этапе сложные органические полимеры (клетчатка, белки, жиры и др.) под действием различных видов анаэробных бактерий разлагаются до более простых соединений, которые затем на втором этапе превращаются метанообраэующими бактериями в метан, углекислый газ и воду. Образованный метан перекачивают в газгольдер, откуда в последствие его берут для дальнейшего использования.

После анаэробного сбраживания в биореакторе остается осадок, так называемый биошлам. Ценность биошлама заключается в том, что содержащиеся в исходной сбраживаемой массе полезные элементы (фосфор, калий и азот) полностью остаются в биошламе, а часть вредных нитратов и нитритов в процессе ферментации сбраживаются в аммиак и метан. Шлам можно разделить на две фракции: жидкую и твердую, каждая из которых является удобрением. Жидкая фаза шлама после анаэробной переработки может сразу же использоваться как удобрение для сельскохозяйственных культур. Твердая фракция шлама может использоваться не только как удобрение, так и в качестве подкормки для крупного рогатого скота.

Таким образом, протекающие в биоэнергетическом комплексе процессы представляет собой экологически чистый, безотходный способ переработки органических отходов животного и растительного происхождения с целью получения биогаза и удобрений, причем такое натуральное биоудобрение полезнее, чем минеральные удобрения, содержащие дополнительные химические добавки.

Системы навозоудаления в биоэнергетическом комплексе. Очевидно, что эффективность работы всего биоэнергетического комплекса зависит от качества работы каждого его элемента. Рассмотрим самый первый этап работы комплекса – процедуру навозоудаления. Именно от является самым трудоемким и технически сложным, на который тратится примерно от 30 до 50% трудовых затрат. Для реализации этой процедуры используется механический способ удаления навоза, обычно посредством системы скребковых навозоуборочных транспортеров. Использование таких транспортеров позволяет механизировать одну из самых сложных задач в животноводческом хозяйстве – уборку навоза крупнорогатого скота в помещениях любого размера, что позволяет значительно экономить трудовые ресурсы [10].

Система навозоудаления на крупных животноводческих фермах содержит различные поточно-транспортные механизмы циклического действия. Эти механизмы по их функциональному назначению можно разделть на три группы:

1 – продольные механизмы (обычно кругового движения), осуществляющие удаление навоза из зон расположения животных; 2 – поперечные механизмы, осуществляющие удаление навоза из помещения; 3 – транспортные механизмы, осуществляющие погрузку навоза в навозосборник и его транспортировку в хранилище.

Обычно система навозоуборочных транспортеров состоит из двух транспортеров – горизонтального и наклонного, каждый из которых имеет собственный приводной двигатель и работает независимо друг от друга. Для примера на рис.2 показана технологическая кинематическая схема системы удаления навоза двухрядного коровника.

Горизонтальный транспортер (ГТ) 1 состоит из кованой замкнутой цепи со скребками, поворотных устройств для изменения направления движения цепи и натяжного

устройства. Цепь приводится в движение от электродвигателя Д1 через редуктор и размещается в навозном канале 3 животноводческого помещения.

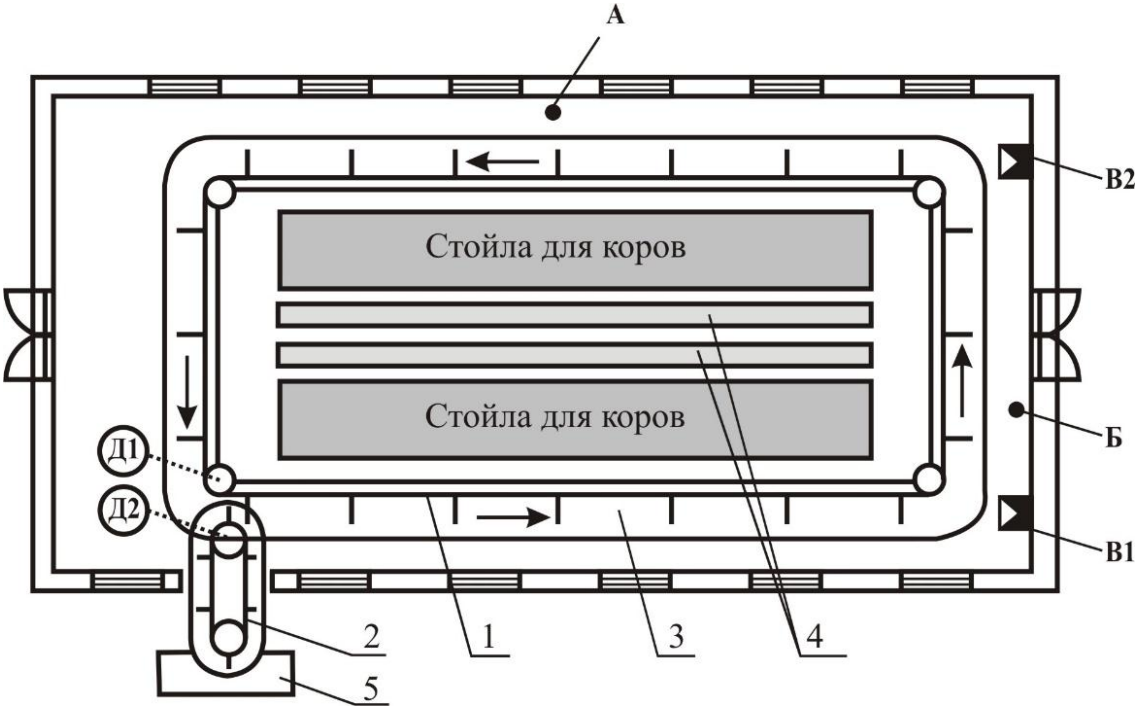


Рисунок 2 – Кинематическая схема системы навозоуборки двухрядного коровника:
1 – горизонтальный транспортер; 2 – наклонный транспортер; 3 – навозный канал;
4 – кормушки; 5 – навозосборник; Д1 и Д2 – приводные электродвигатели
горизонтального и наклонного транспортеров

Наклонный транспортер (НТ) 2 устанавливается под углом к горизонту не более 30^0 и имеет желоб, в котором расположена замкнутая цепь со скребками, которая движется в этом желобе за счет приводного электродвигателя Д2. Нижнюю часть НТ устанавливают под цепью ГТ, а под верхним концом НТ размещают навозосборник 5 для сбора навоза. Приводные электродвигатели Д1 и Д2 ГТ и НТ, а также пульт их управления установлены в зоне перегрузки навоза.

Разработка схемы управления системы навозоуборочных транспортёров. Проведенный анализ показал, что процесс работы навозоуборочных транспортёров характеризуется целым рядом специфических особенностей, которые необходимо учесть при разработке схемы их управления. Среди этих особенностей выделим следующие.

При работе ГТ навоз из навозного канала перемещается скребками цепи на нижнюю часть НТ, расположенную внутри помещения. НТ обеспечивает удаление навоза из помещения, подъем его на заданную высоту и загрузку навоза в навозосборник 5. При этом сбрасывать навоз на неподвижную ветвь НТ нельзя, так как в этом случае при пуске резко перегружаются его цепь и механизмы привода. Поэтому сначала включают НТ, затем ГТ. Выключают транспортеры в обратном порядке.

При работе ГТ нагрузка его приводного электродвигателя Д1 изменяется: при пуске нагрузка имеет максимальное значение, а по мере удаления навоза из навозного канала и перемещения его в приемную часть НТ нагрузка уменьшается. В конце цикла уборки нагрузка уменьшится до ее значения при холостом ходе. Расчеты показывают, что при пуске нагрузка примерно в 3÷4 раза больше, чем в конце уборки. Эту особенность работы ГТ необходимо учитывать при выборе электродвигателя для его привода. Этот выбор осуществляют с учетом требуемого пускового момента и мощности при

максимально возможной нагрузки в начале уборки. Для обеспечения высокой эксплуатационной надежности путем исключения работы при резком увеличении нагрузки на цепи вследствие ее заедания или заклинивания, ГТ должен иметь возможность кратковременно реверсироваться.

Характер изменения нагрузки на НТ иной: его пуск осуществляется вхолостую, затем нагрузка увеличивается до номинальной, после чего постепенно опять снижается до холостого хода. При этом скорость движения цепи НТ должна быть значительно выше, чем ГТ, что обеспечивает выгрузку жидкого навоза. Следует также учесть, что нагрузка на транспортеры при прочих равных условиях зависит от температуры навоза. Кроме того, в зимний период необходимо обеспечить чистоту желоба НТ для предотвращения примерзания его цепи. Для этого необходимо предусмотреть работу НТ вхолостую в течении примерно 4 ÷ 5 минут после остановки ГТ.

Навозоуборочные транспортера работают в достаточно тяжелых окружающих условиях, характеризующихся повышенном содержание вредных газов, влажностью и т.д. Поэтому наряду с выполнением приведенных выше особенностей работы навозоуборочных транспортеров, схема их управления должна обеспечивать также защиту от коротких замыканий, обрыва фаз сети, самопроизвольного пуска (нулевая защита), превышения допустимой температуры, а также обеспечивать оперативное отключение двигателя при его перегрузки, неисправности кинематических элементов всей системы, нахождении в опасной зоне животных и/или людей.

На рис. 3 приведена принципиальная электрическая схема силовой цепи системы навозоуборочных транспортеров.

Приводные электродвигатели Д1 горизонтального и Д2 наклонного транспортеров управляются посредством преобразователей частоты ПЧ-1 и ПЧ-2 соответственно.

Для включения питания и защиты ПЧ (ПЧ-1 и ПЧ-2) и схемы управления от токов короткого замыкания используются автоматические выключатели QF1, QF2 и QF3.

Непосредственное управление режимами работы двигателей Д1 горизонтального и Д2 наклонного транспортеров, а также их защита (тепловая, от потери фаз, от заклинивания ротора и т.д.) обеспечивается частотными преобразователями ПЧ-1 и ПЧ-2.

Для управления самим ПЧ используются следующие его входы:

- GND – общий провод;
- M0 – при соединение которого с GND двигатель начинает вращаться в прямом направлении с заданным в ПЧ темпом ускорения, или осуществляется остановка с заданным в ПЧ темпом замедления;
- M1 – при соединении которого с GND, двигатель начинает вращаться в обратном направлении с заданным в ПЧ темпом ускорения, или осуществляется остановка с заданным в ПЧ темпом замедления если M0 не подтянут к GND;
- M2 – при соединение с GND привод может работать, при размыкании наступает немедленное отключение привода (торможение с выбегом) с блокировкой. После срабатывания, чтоб снять блокировку привода, нужно подать сигнал на M3;
- M3 – выполняет сброс блокировки привода. Сброс произойдет, если нет сигнала аварии M2;
- ACI – для внешнего задания выходной частоты используется аналоговый вход настроенный на 4-20мА.

Для сигнализации внешних цепей ПЧ использует выходы:

- КА1 и КА2 – для сигнализации аварийных состояний (срабатывания защит). Нормально открытые контакты (клеммы RA, RC), соответствующих ПЧ.
- MO1 – для сигнализации работы приводов. Транзисторный выход (клеммы MO1, MCM), соответствующих ПЧ.

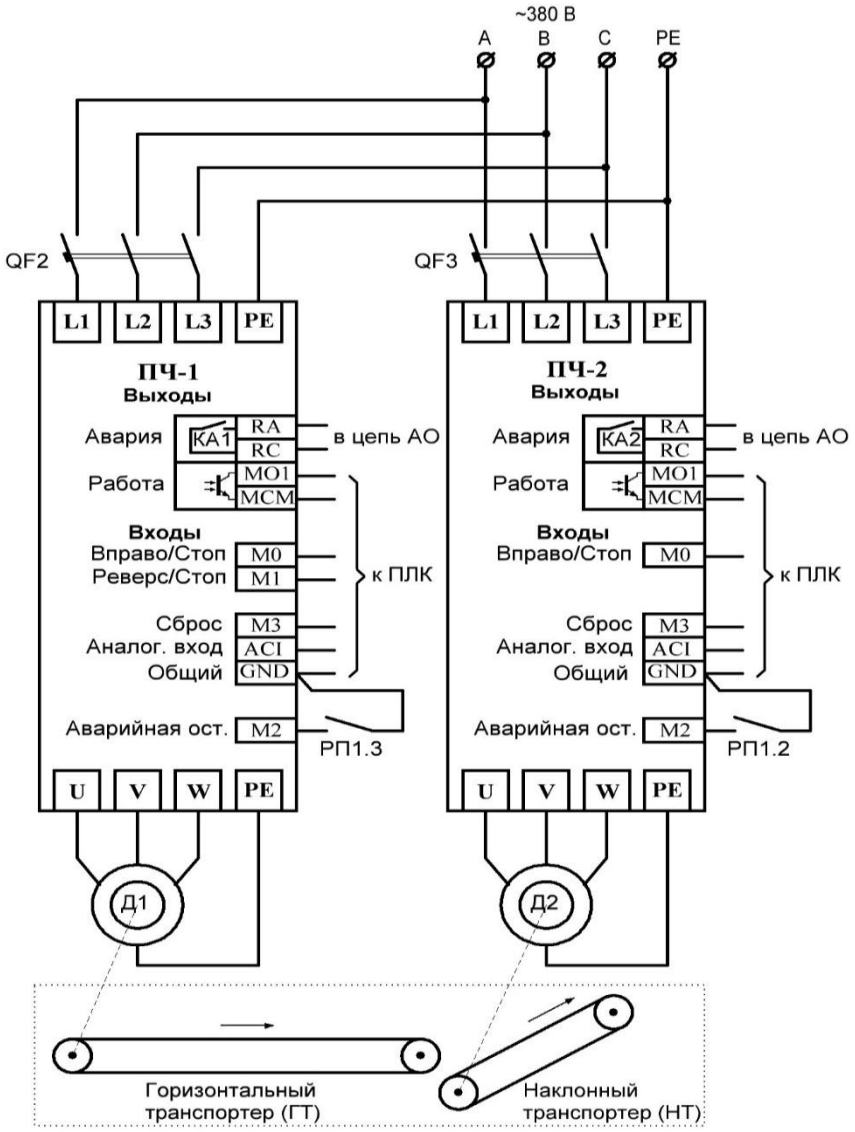


Рисунок 3 – Схема системы управления (силовая цепь)

Логику работы транспортеров, обработку сигналов с датчиков, кнопок, выдачу световых сигнализаций и т.д. выполняет программа, зашитая в программируемый логический контроллер (рис. 4).

Ввод данных в программируемый логический контроллер (ПЛК) осуществляется с помощью его цифровых и аналоговых входов, а выдача результатов с которого осуществляется с помощью его цифровых (релейных) и аналоговых выходов.

Для управления ПЧ-1 используются цифровые релейные выходы DO8, DO9, DO10 и аналоговый выход AO2 (4-20мА) подключенные к соответствующим входам ПЧ-1. Для управления ПЧ-2 используются цифровые релейные выходы DO6, DO7 и аналоговый выход AO1 (4-20мА), подключенные к соответствующим входам ПЧ-2. Для реализации аварийной остановки, и блокировка пуска до готовности всех элементов системы, используется промежуточное реле РП1.

Пуск и остановка электродвигателей Д1 ГТ и Д2 НТ осуществляется при помощи кнопок SB2, SB2′, SB2′′ и SB3, причем кнопки SB2 и SB3 располагаются в шкафу управления, а кнопки SB2′ и SB2′′ – в других местах коровника, удаленных от шкафа управления.

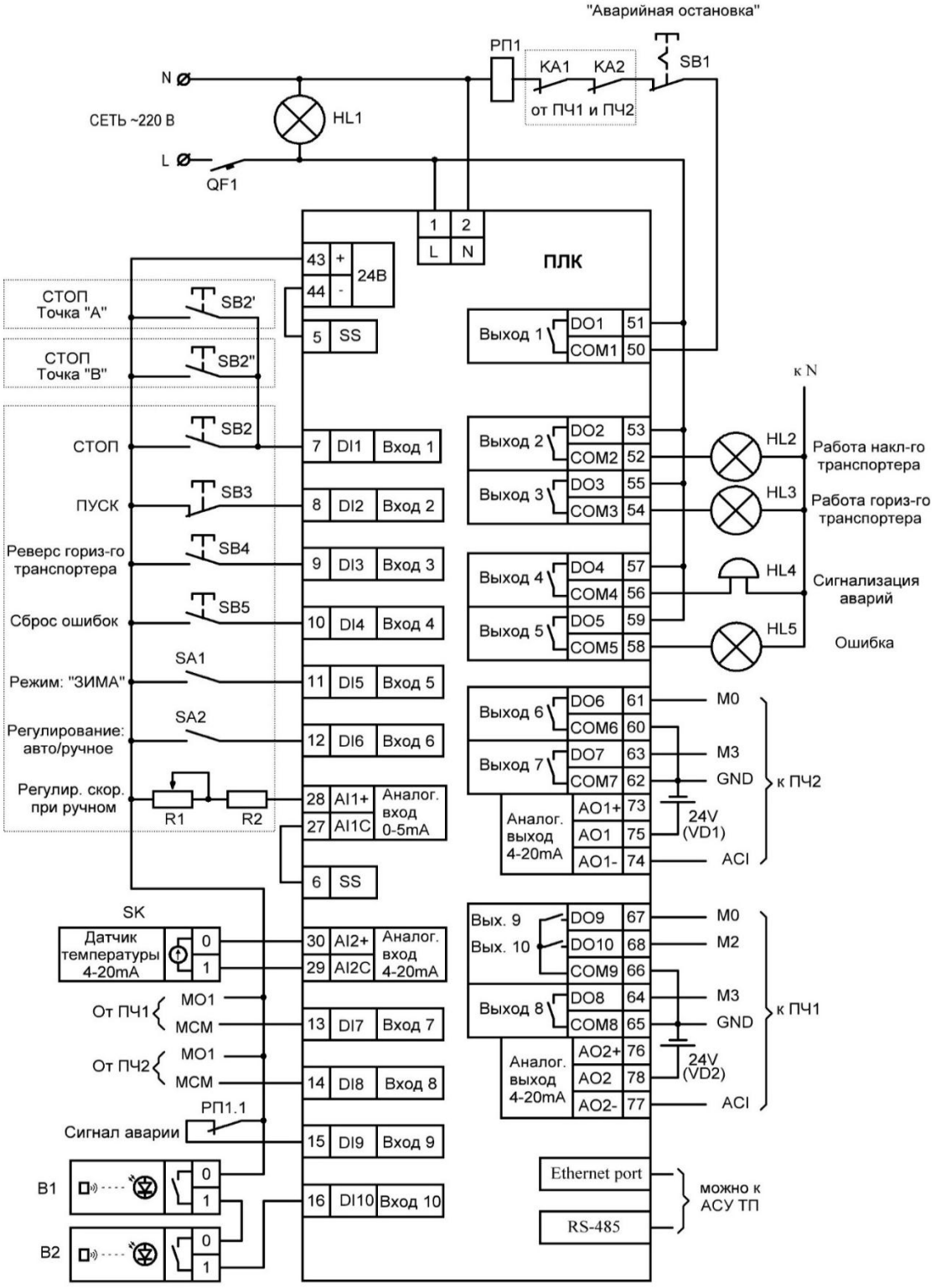


Рисунок 4 – Схема системы управления (цепи управления)

При нажатии кнопки SB3 ПЛК подает команду на включение системы транспортеров по заданному алгоритму работы. При этом программа ПЛК не допустит включения ГТ в прямом направлении при отключенном НТ. При запуске сначала запускается НТ, после разгона которого включается ГТ. Поскольку пуск ГТ требует большого пускового момента, то ПЧ-1, управляющий Д1, настроен на векторное управление, поскольку векторное управление дает высокий момент во время пуска вплоть до выхода

на номинальный режим. Можно также использовать скалярное управление с форсировкой момента.

Для кратковременного реверса ГТ предусмотрено кнопка SB4, причем алгоритм ПЛК для реверса обеспечивает работу в двух режимах:

- если транспортеры работают: при зажатой SB4, ГТ будет замедляться, до скорости равным нулю (темп замедления настраивается в ПЧ-1), после чего начнет вращать в противоположную сторону. Теперь если отпустить ТГ снизит скорость до нуля и начнет вращаться в правильном направлении, при этом НТ как работал, так и работает.

- если транспортеры не работают: при нажатой SB4, ГТ начнет вращаться в обратном направлении и остановится после отпускания кнопки SB4.

Остановка транспортеров осуществляется посредством кнопки SB2. Для возможности оперативной аварийной остановки приводов транспортеров при нахождении оператора вдали от пульта управления кнопка “СТОП” SB2 дважды продублирована двумя кнопками SB2′ и SB2′′, которые располагаются, например, в центре коровника и в противоположном от места расположения электродвигателей Д1 и Д2 торце коровника (точки А и Б на рис 2).

Для сброса ошибок после аварийной остановки предусмотрено кнопка SB5.

Для предотвращения примерзания цепи и скребков наклонного транспортера в зимнее время предусмотрен тумблер SA1 включения режима “Зима”, при включении которого алгоритм ПЛК, после нажатия кнопки “СТОП” SB2, останавливает ГТ и только через 5 минут останавливает НТ. При этом, если НТ еще не остановился, при необходимости возможен повторный запуск ГТ.

Для выбора способа регулирования скорости (ручной или автоматический) предусмотрен тумблер SA2. При отсутствии автоматического управления для подачи сигнала задания скорости в ПЛК предусмотрен переменный резистор R1, подключенный к аналоговому входу AI1 (ПЛК) через токоограничивающий резистор R2. Частота вращения приводов ГТ и НТ может регулироваться от 30% до номинальной.

Для предотвращения повреждения НТ, программа ПЛК не допустит запуск при температуре ниже нормы. Данные о температуре окружающей среды поступают с температуры SK, подключенного к аналоговому входу ПЛК. Норма температуры задана в программе ПЛК. Если все-таки транспортер необходимо включить, то включится сирена без сигнала ошибки HL5. Если температура навоза увеличивается под воздействием температуры наружного воздуха, то при включенном тумблере SA2, т.е. при “автоматическом” управлении, программа ПЛК изменит скорость ГТ и НТ согласно заданной в программе зависимости $n = f(t^{o})$ путем подачи сигналов на аналоговый вход ПЧ-1 и ПЧ-2. В случае приближения человека и/или животного к опасной зоне, датчики движения B1 и B2 (см. рис 2), подключенные ко входу DI9, подают в ПЛК сигнал об этом. Далее, согласно программе, ПЛК включает сирену, размыкает релейный “выход 1” и питание промежуточного реле РП1 пропадает. Тем самым размыкаются контакты РП1.1, РП1.2 и РП1.3 и два последних подают сигнал аварийной остановки ПЧ-1 и ПЧ-2 с блокировкой приводов. Аварийная остановка производится на свободном выбеге. Сигналы работы для ПЧ при этом снимаются. Повторный запуск возможен после удаления из опасной зоны человека/животного после двойного нажатия кнопки SB5 сброса ошибки.

Для внешней аварийной остановки приводов в схеме управления предусмотрено фиксирующая кнопка аварийной остановки SB1, включенная в цепь аварийной остановки, при нажатии на которую останавливается работа приводов с невозможностью повторного включения (вне зависимости от других частей схем управления). Для сигнализации аварий, вызванных неисправностью ПЧ-1, ПЧ-2 и кнопки аварийной остановки SB1, к цифровому входу DI9 подключен нормально закрытый контакт РП1.1 промежуточного реле РП1.

Для визуальной сигнализации состояния всей систем предусмотрен ряд световых индикаторов: HL1, указывающий на наличие напряжения в сети; HL2 и HL3, сигнализирующие о работе НТ и ГТ соответственно; HL5 для индикации ошибок. Кроме того, предусмотрена сирена HL4 для сигнализации при возникновении аварий.

Следует отметить, что аварийную остановку вызывают также следующие причины:

• нажатие на аварийную кнопку;

• неисправность любого из ПЧ, а также срабатывание защиты двигателей (тепловая, КЗ, заклинивание, потери фаз и т.д.), QF2, QF3. Информация о всем этом получается через контакты ПЧ, а именно КА1 и КА2;

• по инициативе ПЛК;

• при выключенном цепи управления, т.е. QF1.

Условиями отсутствия аварий является:

- наличие питания всех цепей, т.е. QF1, QF2 и QF3 включены;
- ПЧ-1 включен и готов к работе (нормально открытые контакты КА1 “ЗАМКНУТО”);
- ПЧ-2 включен и готов к работе (нормально открытые контакты КА2 “ЗАМКНУТО”);
- кнопка аварийной остановки SB1 замкнута (нормальное состояние);
- ПЛК готов к работе (релейный выход 1 замкнут);
- в опасной зоне нет животных и людей, а температура не ниже предельно допустимой.

Нарушение хотя бы одного из перечисленных условий приводит к аварийной остановке приводов ГТ и НТ и блокировки приводов от включения путем размыкания контактов РП1.2 и РП1.3 подключенных к ПЧ-1 и ПЧ-2.

В процессе работы программа, зашитая в ПЛК, выполняет следующие операции:

1. В цикле опрашивает состояние входов DI1, DI2, DI3 и DI4 для определения нажатия кнопок и при нажатии выполняет соответствующие действия.

2. В цикле опрашивает состояние входа DI6 (авто/руч.), а также аналоговые входы AI1 и AI2 на основании которых выполняет следующие операции:

- в случае, если выбран ручное регулирование, то в переменную частоты записывается значение аналогового входа AI1, естественно умноженному на определенный коэффициент пропорциональности, с проверкой корректности;

- в случае автоматического управления в переменную частоты поддержания записывается значение взятой из зависимости $f=f(t^o)$ значение температуры берется с входа AI2, к которому подключен датчик температуры SK. Также на основании температуры по SK определяется нижний порог температуры, при достижении которого выполняет функцию аварийной остановки только без включения сирены, но с блокировкой приводов. Сбросить ошибку нельзя до тех пор, пока температура не станет допустимой для работы.

3. Каждый 0,5 секунды по DO6, DO9 и DO10 проверяет, поданы ли сигналы (выходы M0, M1) в оба ПЧ. При наличии хотя бы одного, подает на аналоговые выходы AO1 и AO2 ток, пропорциональный значению переменной частоты поддержания. При этом в момент нажатия кнопок пуск или реверс, на аналоговые выходы сразу подается ток, пропорциональный значению переменной частоты поддержания, который в этом цикле при необходимости изменяется и корректируется при работе.

4. В цикле проверяет вход от аварийной остановки DI9, к которому подключен контакт РП1.1, а также вход DI10, к которому подключен датчики движения. В обоих случаях дает сигнал аварийную остановку.

Для возможности модернизации и подключения системы навозоудаления к АСУ ТП биоэнергетического комплекса, в ПЛК предусмотрены интерфейсы ethernet и RS-485.

Экспериментальная проверка, проведенная в лабораторных условиях с использованием системы, содержащей два асинхронных электродвигателя типа 4А80В6, два частотных преобразователя Delta VFD-022M и программируемый логический контроллер типа ПЛК160 (МО2), подтвердила работоспособность и эффективность разработанной схемы управления. Это позволяет рекомендовать ее к практическому применению при создании автоматизированных биоэнергетических комплексов по переработке органических отходов животного происхождения с целью получения из них биогаза и удобрений.

Заключение. Предложенная схема управления электроприводом системы навозоудаления, входящей в состав биоэнергетического комплекса, полностью обеспечивает выполнение всех требований, предъявляемых к совместной работе горизонтального и наклонного транспортеров. Регулирование приводными электродвигателями осуществляется посредством преобразователей частоты, управляемых программируемым логическим контроллером, для которых разработан алгоритм работы. При этом следует отметить, что разработанная схема управления устраняет рывки и удары при изменении режимов работы транспортеров, что повышает надежность и долговечность функционирования транспортеров.

При необходимости, разработанную схему можно увязать с работой всех остальных структурных составляющих этого биоэнергетического комплекса, организовав тем самым обобщенную АСУ ТП.

Литература

1. Регламент (ЕС) № 1069/2009 Европейского парламента и Совета Европейского союза «Санитарные нормы в отношении побочных продуктов животного происхождения и производных продуктов, не предназначенных для потребления человеком», 21 октября 2009 г.
2. Сидоренко О.Д. Биоконверсия отходов агропромышленного комплекса. – М.: Инфра-М, 2018. – 160 с.
3. Сидоренко О.Д., Борисенко Е.Г., Ванькова А.А., Войно Л.И. Микробиология. – М.: НИЦ Инфра-М, 2016. – 286 с.
4. Егорова Т.А., Клунова С.М., Живухина Е.А. Основы биотехнологии. – М.: Академия, 2003. – 208 с.
5. Шукуров У.Ш., Акчалов Ш. Возобновляемые источники энергии в Кыргызской Республике: состояние, проблемы и перспективы // Проблемы автоматики и управления. – 2015. № 1 (28). – С. 72-76.
6. Шомин А.А. Биогаз на сельском подворье. – М.: Балаклея, 2002. – 68 с.
7. Обозов А. Д. Возобновляемые источники энергии - основа экологической и экономической безопасности // Известия КГТУ, 2012, № 26. – С. 182-192.
8. Баадер В., Доне Е. Биогаз: теория и практика. – М.: Колос, 1982. – 148 с.
9. Добышев А.С., Пузевич К.Л., Острейко А.А. Подготовка смесей органического сырья для производства биогаза // Современные проблемы освоения новой техники, технологий, организации технического сервиса в АПК: доклады Междунар. научно-практическая конференция. – Минск: БГАТУ, 2013. – С 202-205.
10. Павлов П.И. Эффективные средства механизации для удаления и утилизации навоза // Естественные и технические науки. – 2017, №3 (105). – С. 87-89.

УДК 510.67 10.5281/zenodo.3904101

А.Т.Нуртазин, З.Г. Хисамиев. *KhisamievZ@mail.ru*
Институт информационных и вычислительных технологий КН МОН РК, Казахстан

ЭКЗИСТЕНЦИАЛЬНО ЗАМКНУТЫЕ КОМПАНЬОНЫ КОЛЬЦА ЦЕЛЫХ ЧИСЕЛ

В работе изучаются компаньон-модели кольца целых чисел. Это исследование является примером изучения классического объекта посредством теории классов Фрэсе, разработанной Нуртазиным А.Т. В исследовании описаны структуры компаньонов и экзистенциально замкнутых компаньонов кольца целых чисел.

Ключевые слова: компаньон; экзистенциально замкнутое кольцо; изоморфизм колец; алгебраический элемент; трансцендентный элемент.

Введение

Теория экзистенциальной замкнутости возникла в середине двадцатого столетия в трудах одного из признанных классиков теории моделей Абрахама Робинсона [1, 2], а также в работах [3–8]. В настоящее время она является одной из значительных и наиболее развитых областей современной теории моделей. В предыдущих исследованиях вводится и исследуется наиболее базисная форма широко известного в теории экзистенциальной замкнутости понятия компаньон-теории. Был найден критерий счётной категоричности данной компаньон-теории. Изучены некоторые свойства экзистенциально замкнутых и форсинг-компаньонов [3–9]. Другой перспективный подход построения теории экзистенциально замкнутых структур, основанный на работах Фрэсе [6], развивается в [9–16].

Естественно, что развитие общей теории экзистенциональных замкнутых компаньонов должно сопровождаться исследованием классических структур и теорий. Исторически одним из наиболее классических математических объектов является кольцо целых чисел. В работе строится теория алгебраических компаньон-расширений кольца целых чисел. Это исследование является примером изучения классического объекта посредством подхода, разработанного Нуртазиным А.Т. и основанного на классах Фрэсе. В исследовании строятся экзистенциально замкнутые компаньоны кольца целых чисел и найдено их число с точностью до изоморфизма, оставляющего множество всех трансцендентных над кольцом целых чисел элементов на месте.

1. Модели элементарной теории компаньонов кольца целых чисел

Пусть: $\mathbb{Z}$ – кольцо целых чисел; $\mathbb{Z}[\overline{x}]$ – кольцо многочленов от переменных $\overline{x}=(x_1,...,x_n)$. Все используемые и неприведенные определения, а также обозначения взяты из монографии [15]. Класс компаньонов кольца $\mathbb{Z}[\overline{x}]$ обозначаем $C(\mathbb{Z})$.

Приведем без доказательств несколько результатов из [17]. Эти результаты начальные и будут использованы в настоящей статье.

ТЕОРЕМА 1 [17]. Кольцо целостности $\mathbb{Z}[\overline{x}]$ является компаньоном кольца $\mathbb{Z}$, т.е. $\mathbb{Z}[\overline{x}] \in C(\mathbb{Z})$.

ПРЕДЛОЖЕНИЕ 1 [17]. Пусть $g_i(\overline{x}), h_j(\overline{x}) \in \mathbb{Z}[\overline{x}]$. Система $\&_i\, g_i(\overline{x}) = 0 \,\&\, \&_j\, h_j(\overline{x}) \neq 0$ эквивалентна в $\mathbb{Z}[\overline{x}]$ системе из одного уравнения и одного неравенства $g(\overline{x}) = 0 \,\&\, h(\overline{x}) \neq 0$.

Пусть $A_f = \{(\overline{a}, a) \mid f(\overline{a}, a) = 0,\ (\overline{a}, a) \in Z\}$ – аннулятор f, где $f(\overline{x}, x) \in \mathbb{Z}[\overline{x}, x]$.

ПРЕДЛОЖЕНИЕ 2 [17]. Пусть $I(A_f) = \{\, g \mid g \in \mathbb{Z}[\overline{x}, x],\ A_f \subseteq A_g \,\}$, где неприводимый над кольцом $\mathbb{Z}$ многочлен $f \in \mathbb{Z}[\overline{x}, x] \setminus \mathbb{Z}[\overline{x}]$ содержащий единицу. Тогда, $I(A_f)$ является главным идеалом в кольце $\mathbb{Z}[\overline{x}, x]$ и $I(A_f) = (f)$.

ТЕОРЕМА 2 [17]. Пусть $f(\overline{x}, x)$ – неприводимый и содержащий единицу многочлен над кольцом $\mathbb{Z}$. Тогда алгебраическое расширение $\mathbb{Z}[\overline{x}]/f$ кольца $\mathbb{Z}[\overline{x}]$ является компаньоном $\mathbb{Z}$ если, и только если, аннулятор A_f является бесконечным множеством.

ТЕОРЕМА 3 [17]. Пусть $\mathbb{Z}[\overline{x}, y]/f$ и $\mathbb{Z}[\overline{x}, y]/f/g$ (здесь $\mathbb{Z}[\overline{x}, y]/f/g = (\mathbb{Z}[\overline{x}, y]/f)[z]/g$) – простые алгебраические расширения колец $\mathbb{Z}[\overline{x}]$ и $\mathbb{Z}[\overline{x}, y]/f$ соответственно, $f(\overline{x}, y), g(\overline{x}, y, z)$ – неприводимые многочлены над кольцами $\mathbb{Z}[\overline{x}]$ и $\mathbb{Z}[\overline{x}, y]/f$ соответственно. Тогда алгебраическое расширение $\mathbb{Z}[\overline{x}, y]/f/g$ кольца $\mathbb{Z}[\overline{x}, y]/f$ является компаньоном $\mathbb{Z}$ если и только если множество $A_f \times Z \cap A_g$ бесконечно.

Пусть $B = \{\beta_1, \beta_2, ...\}$, $X = \{x_1, x_2, ...\}$ – счетные множества независимых переменных: $\overline{\beta} = (\beta_1, ..., \beta_m)$, $\overline{x}_n = (x_1, ..., x_n)$, $\overline{f}_n = (f_1, ..., f_n)$, где $f_i(\overline{\beta}, \overline{x}_{i-1}^{\overline{f}_{i-1}}, x_i)$ – неприводимые в кольце $\mathbb{Z}[\overline{\beta}](\overline{x}_{i-1}^{\overline{f}_{i-1}})[x_i]$, где $\overline{x}_i^{\overline{f}_i} = (x_1^{f_1}, ..., x_i^{f_i})$, $x_i^{f_i}$ – корень многочлена $f_i(\overline{\beta}, \overline{x}_{i-1}^{\overline{f}_{i-1}}, x_i)$ в кольце $\mathbb{Z}[\overline{\beta}](\overline{x}_{i-1}^{\overline{f}_{i-1}})[x_i]$. Положим $\mathbb{Z}[\overline{\beta}](\overline{x}_1^{\overline{f}_1}) = \mathbb{Z}[\overline{\beta}, x_1]/f_1(\overline{\beta}, x_1)$, $\mathbb{Z}[\overline{\beta}, x_1]$, $\mathbb{Z}[\overline{\beta}](\overline{x}_2^{\overline{f}_2}) = \mathbb{Z}[\overline{\beta}](\overline{x}_1^{\overline{f}_1})[x_2]/f_2(\overline{\beta}, \overline{x}_1^{\overline{f}_1}, x_2)$ и $\mathbb{Z}[\overline{\beta}](\overline{x}_1^{\overline{f}_1})[x_2]$. По индукции определим $\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n}) = \mathbb{Z}[\overline{\beta}](\overline{x}_{n-1}^{\overline{f}_{n-1}})[x_n]/f_n(\overline{\beta}, \overline{x}_{n-1}^{\overline{f}_{n-1}}, x_n)$ и $\mathbb{Z}[\overline{\beta}](\overline{x}_{n-1}^{\overline{f}_{n-1}})[x_n]$. Таким образом в последовательности $\mathbb{Z}[\overline{\beta}](\overline{x}_1^{\overline{f}_1}), \mathbb{Z}[\overline{\beta}](\overline{x}_2^{\overline{f}_2}), ..., \mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})$ – каждый последующий член $\mathbb{Z}[\overline{\beta}](\overline{x}_{i+1}^{\overline{f}_{i+1}})$ есть простое алгебраическое расширение предыдущего члена посредством неприводимого многочлена $f_{i+1}(\overline{\beta}, \overline{x}_i^{\overline{f}_i}, x_{i+1})$.

ТЕОРЕМА 4. Пусть в $g(\overline{\beta},\overline{x}_n^{\overline{f}_n})=h(\overline{\beta},\overline{x}_n^{\overline{f}_n})$ выполнено в кольце $\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})$, полученном посредством последовательных присоединений корней $x_1^{f_1},...,x_n^{f_n}$ неприводимых многочленов $f_1,...,f_n$ в соответствующих кольцах $\mathbb{Z}[\overline{\beta}](\overline{x}_{i-1}^{\overline{f}_{i-1}})[x_i]$ $\mathbb{Z}[\overline{\beta}]\subseteq\mathbb{Z}[\overline{\beta}](x_1^{f_1})\subseteq\mathbb{Z}[\overline{\beta}](\overline{x}_2^{\overline{f}_2})\subseteq...\subseteq\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})$.

Тогда в кольце $\mathbb{Z}[\overline{\beta},\overline{x}_n)$ выполняется соотнешение $g(\overline{\beta},\overline{x}_n)=h(\overline{\beta},\overline{x}_n)+q_1(\overline{\beta},\overline{x}_n)f_1(\overline{\beta},\overline{x}_1)+q_n(\overline{\beta},\overline{x}_n)f_2(\overline{\beta},\overline{x}_2)+...+q_n(\overline{\beta},\overline{x}_n)f_n(\overline{\beta},\overline{x}_n)$ для подходящих $q_1(\overline{\beta},\overline{x}_n),...,q_n(\overline{\beta},\overline{x}_n)\in\mathbb{Z}[\overline{\beta},\overline{x}_n]$.

Доказательство индукции по n. Базис индукции n=1. В фактор-кольце $\mathbb{Z}[\overline{\beta}](\overline{x}_1^{\overline{f}_1})=\mathbb{Z}[\overline{\beta}](x_1^{f_1})$ из равенства $g(\overline{\beta},\overline{x}_n^{\overline{f}_n})=h(\overline{\beta},\overline{x}_n^{\overline{f}_n})$ следует равенство $g(\overline{\beta},x_1^{f_1})=h(\overline{\beta},x_1^{f_1})+q_1(\overline{\beta},x_1)f_1(\overline{\beta},x_1)$ для подходящего многочлена $q_1(\overline{\beta},x_1)\in\mathbb{Z}[\overline{\beta},x_1]$. Индуктивный переход. Из равенства $g(\overline{\beta},\overline{x}_n^{\overline{f}_n})=h(\overline{\beta},\overline{x}_n^{\overline{f}_n})$ следует равенство $g(\overline{\beta},\overline{x}_{n-1}^{\overline{f}_{n-1}},x_n)=h(\overline{\beta},\overline{x}_{n-1}^{\overline{f}_{n-1}},x_n)+q_n(\overline{\beta},\overline{x}_{n-1}^{\overline{f}_{n-1}},x_n)f_n(\overline{\beta},\overline{x}_{n-1}^{\overline{f}_{n-1}},x_n)$. В последнем равенстве число алгебраических элементов равно $n-1$, при этом независимые переменные – это $\overline{\beta},x_n$. По предположению индукции последнее соотношение в кольце $\mathbb{Z}[\overline{\beta},x_n](\overline{x}_{n-1}^{\overline{f}_{n-1}})$ влечет соотношение (1):

$$\begin{aligned}&g(\overline{\beta},\overline{x}_n)=h(\overline{\beta},\overline{x}_n)+q_n(\overline{\beta},\overline{x}_n)f_n(\overline{\beta},\overline{x}_n)+q_1(\overline{\beta},\overline{x}_n)f_1(\overline{\beta},\overline{x}_1)+\\&+q_n(\overline{\beta},\overline{x}_n)f_2(\overline{\beta},\overline{x}_2)+...+q_{n-1}(\overline{\beta},\overline{x}_n)f_{n-1}(\overline{\beta},\overline{x}_{n-1}).\end{aligned}\qquad(1)$$

Теорема доказана.

ТЕОРЕМА 5. Кольцо $\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})$ является компаньоном кольца $\mathbb{Z}$ тогда и только тогда, когда аннулятор $A_{\overline{f}_n}=A_{f_1}\times Z^{n-1}\cap A_{f_2}\times Z^{n-2}\cap...\cap A_{f_n}$ является бесконечным множеством.

Доказательство. Необходимость. Допустим противное – $A_{\overline{f}_n}=\{(\overline{b}^1,\overline{a}^1),...,(\overline{b}^k,\overline{a}^k)\}$, где $(\overline{b}^i,\overline{a}^i)=(b_1^i,...,b_m^i,a_1^i,...,a_n^i)\in Z$. Тогда выполнено: $\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})\models\exists u_1...u_m\exists v_1...v_n(f_1(\overline{u},\overline{v}_1)\cdot...\cdot f_n(\overline{u},\overline{v}_n)=0\,\&\,\&_i(\overline{u},\overline{v})\neq(\overline{b}^i,a^i))$,

где $\overline{u}=(u_1,...,u_m),\overline{v}=(v_1,...,v_n),\overline{v}_i=(v_1,...,v_i)$. Так как $\mathbb{Z}$ и $\mathbb{Z}[\overline{\beta}](\overline{x}_n^{\overline{f}_n})$ компаньоны, а параметры $(\overline{b}^1,\overline{a}^1),...,(\overline{b}^k,\overline{a}^k)$ Е-выразимы в $\mathbb{Z}$, то в $\mathbb{Z}$ имеется решение $f_1(\overline{u},\overline{v}_1)\cdot...\cdot f_n(\overline{u},\overline{v}_n)=0$ не принадлежащее A_f. Противоречие. Необходимость доказана.

Достаточность. Вначале докажем следующее предложение.

ПРЕДЛОЖЕНИЕ 3. Пусть $I(A_{\bar{f}_n})=\{\ g \mid g \in \mathbb{Z}[\bar{\beta},\bar{x}_n],\ A_{\bar{f}_n} \subseteq A_g, \bar{f}_n=(f_1,...,f_n)\}$, где каждый неприводимый в кольце $\mathbb{Z}[\bar{\beta},x_i](\bar{x}_{i-1}^{\bar{f}_{i-1}})$ многочлен $f_i(\bar{\beta},\bar{x}_{i-1},x_i), i=1,..,n$ содержит единицу. Тогда идеал $I(A_{\bar{f}_n})$ в кольце $\mathbb{Z}[\bar{\beta},\bar{x}_n]$ порождается многочленами $f_1,...,f_n$ т.е. $I(A_{\bar{f}_n})=(\bar{f}_n)$.

Доказательство. Действительно, неприводимый многочлен $f_{n-1}(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)$ в кольце целостности $\mathbb{Z}[\bar{\beta},x_{n-1}](\bar{x}_{n-1}^{\bar{f}_{n-1}})$ является также неприводимым многочленом в кольце целостности $\widetilde{\mathbb{Z}[\bar{\beta},x_{n-1}](\bar{x}_{n-1}^{\bar{f}_{n-1}})}[x_n]$ над полем частных $\widetilde{\mathbb{Z}[\bar{\beta},x_{n-1}](\bar{x}_{n-1}^{\bar{f}_{n-1}})}$. В $\widetilde{\mathbb{Z}[\bar{\beta},x_{n-1}](\bar{x}_{n-1}^{\bar{f}_{n-1}})}[x_n]$ идеал $I(\widetilde{A_{\bar{f}_n}})$ будет главным, порождающим элементом которого будет неприводимый многочлен $f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)\in I(\widetilde{A_{\bar{f}_n}})$. Пусть $g\in I(A_{\bar{f}_n})\subseteq I(\widetilde{A_{\bar{f}_n}})$, тогда $f_n \mid g$ (f_n делит g) в кольце $\widetilde{\mathbb{Z}[\bar{\beta}](\bar{x}_{n-1}^{\bar{f}_{n-1}})}[x_n]$, отсюда следует, что выполнено равенство

$$n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}})g(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)=m(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}})f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)q(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n),$$

где $n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}}),m(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}})\in\mathbb{Z}[\bar{\beta}](\bar{x}_{n-1}^{\bar{f}_{n-1}}),q(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)\in\mathbb{Z}[\bar{\beta}](\bar{x}_{n-1}^{\bar{f}_{n-1}})[x_n]$. В силу однозначности разложения в кольце целостнсти на неприводимые множители имеем $f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)\mid n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}})$ или $f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)\mid g(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)$. Очевидно первое невозможно вследствие выбора $f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)$. Следовательно $f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)\mid g(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)$. Из теоремы 4 следует, что последнее соотношение делимости $g(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)=f_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)q_n(\bar{\beta},\bar{x}_{n-1}^{\bar{f}_{n-1}},x_n)$ в кольце $\mathbb{Z}[\bar{\beta}](\bar{x}_{n-1}^{\bar{f}_{n-1}})[x_n]$ влечет соотношение $g(\bar{\beta},\bar{x}_n)=f_n(\bar{\beta},\bar{x}_n)q_n(\bar{\beta},\bar{x}_n)+f_{n-1}(\bar{\beta},\bar{x}_{n-1})q_{n-1}(\bar{\beta},\bar{x}_n)+...+f_{n-1}(\bar{\beta},x_1)q_{n-1}(\bar{\beta},x_1)\ \mathbb{Z}[\bar{\beta},\bar{x}_n]$, т.е. $I(A_{\bar{f}_n})=(\bar{f}_n)$. Предложение доказано.

Теперь вернёмся к доказательству достаточности. Пусть $A_{\bar{f}_n}=A_{f_1}\times Z^{n-1}\cap A_{f_2}\times Z^{n-2}\cap...\cap A_{f_n}$ бесконечное множество. Достаточно доказать, что в моделях $\mathbb{Z}$ и $\mathbb{Z}[\bar{\beta}](\bar{x}_n^{\bar{f}_n})$ выполняются одни и те же экзистенциальные предложения. Очевидно, что экзистенциальное предложение, истинное в $\mathbb{Z}$, также истинно в $\mathbb{Z}[\bar{\beta}](\bar{x}_n^{\bar{f}_n})$. Докажем, что каждое экзистенциальное предложение φ, выполненное в $\mathbb{Z}[\bar{\beta}](\bar{x}_n^{\bar{f}_n})$, выполняется в $\mathbb{Z}$. Можно считать, что бесквантворная формула φ, содержащая решения в кольце целлостности $\mathbb{Z}[\bar{\beta}](\bar{x}_n^{\bar{f}_n})$, после подстановки этих решений принимает вид $\&g_i(\bar{\beta},\bar{x}_n^{\bar{f}_n})=0\ \&\ h(\bar{\beta},\bar{x}_n^{\bar{f}_n})\neq 0$. В последнем неравенстве мы заменили несколько неравенств их произведением. Из предложения 3 следует, что $g_i(\bar{\beta},\bar{x}_n)\in I(A_{\bar{f}_n}), h(\bar{\beta},\bar{x}_n)\notin I(A_{\bar{f}_n}), i=1,...,n.$ Следовательно, $A_{\bar{f}_n}\setminus A_h\neq\varnothing$. Для $\bar{p}_m,\bar{q}_n\in A_{\bar{f}_n}\setminus A_h$ в кольце $\mathbb{Z}$ выполняется система $\&g_i(\bar{p}_m,\bar{q}_n)=0\&h(\bar{p}_m,\bar{q}_n)\neq 0$, что завершает доказательство достаточности, а вместе с этим и доказательство теоремы 5.

2 Экзистенциально замкнутые модели кольца целых чисел и их число

Укажем алгоритм построения экзистенциально замкнутых моделей.

Пусть $F=\{f_1 f_2,...\}$, где $f_i(\overline{\beta}_i,x_i)$ – неприводимы над кольцом $\mathbb{Z}$, $\overline{\beta}_j \in B, x_k \in X$, $B=\{\beta_1,\beta_2,....\}$, $X=\{x_1,x_2,....\}$, $B=\{\beta_1,\beta_2,....\}$, $X=\{x_1,x_2,....\}$ – счетные множества независимых переменных и все переменные из B в многочлене $f_i(\overline{\beta}_i,x_i)$ содержатся среди $\overline{\beta}_i$. Шаг 1. Положим $\mathbb{Z}_1^F[B]=\mathbb{Z}[B]/f_1$. Так как экзистенциальное предложение выполненное в $\mathbb{Z}_1^F[B]$ выполняется в подкольце $\mathbb{Z}_1^F[\overline{\beta}_s]\in C(\mathbb{Z})$ для некоторого s, то $\mathbb{Z}_1^F[B]\in C(\mathbb{Z})$. Пусть уже определены компаньоны $\mathbb{Z}_1^F[B]\subseteq...\subseteq\mathbb{Z}_{n-1}^F[B]$. Шаг n. Положим $\mathbb{Z}_1^F[B]=\mathbb{Z}_{n-1}^F[B]/f_n^*$, где f_n^* неприводимый над кольцом $\mathbb{Z}_n^F[B]$ делитель первого неиспользованного многочлена $f_{n^*}(\overline{\beta}_{n^*},x_{n^*})$ из $F=\{f_1,f_2,...\}$, неимеющий ни одного корня в $\mathbb{Z}_{n-1}^F[B]$ и аннулятор $A_{\overline{f}_n^*}=A_{f_1}\times Z^{n-1}\cap A_{f_2*}\times Z^{n-2}\cap...\cap A_{f_n*}$ имеет бесконечно много нулей. Положим $\mathbb{Z}_\omega^F[B]=\cup\mathbb{Z}_\omega^F[B]$.

ТЕОРЕМА 6. Кольцо целостности $\mathbb{Z}_\omega^F[B]$ является экзистенциально замкнутым компаньоном кольца $\mathbb{Z}$.

Доказательство. Из того, что в возрастающей цепочке колец $\mathbb{Z}_1^F[B]\subseteq...\subseteq\mathbb{Z}_{n-1}^F[B]\subseteq...$ каждое кольцо является компаньоном $\mathbb{Z}$, следует, что и $\mathbb{Z}_\omega^F[B]=\cup\mathbb{Z}_i^F[B]$ является компаньоном $\mathbb{Z}$. Пусть $\mathbb{A}\supseteq\mathbb{Z}_\omega^F[B]$. Докажем, что каждое экзистенциальное предложение φ с параметрами из $\mathbb{Z}^F{}_\omega[B]$, выполненное в $\mathbb{A}$, выполняется в $\mathbb{Z}_\omega[B]$. Можно считать, что бесквантворная формула φ, содержащяя решение в кольце $\mathbb{A}$, после подстановки существующих решений имеет вид $\&g_i(\overline{\beta}_m,\overline{\alpha}_n,\overline{y}_k^{\overline{e}_k})=0\&h(\overline{\beta}_m,\overline{\alpha}_n,\overline{y}_k^{\overline{e}_k})\neq 0$, где $\overline{\alpha}_n=(\alpha_1,...,\alpha_n)$ – новые трансцендентные элементы из $\mathbb{A}$. Докажем, что подкольцо $\mathbb{Z}[\overline{\beta}_m,\overline{\alpha}_n](\overline{x}_k^{\overline{e}_k})$ изоморфна некоторому подкольцу $\mathbb{Z}_\omega^F[B]=\cup\mathbb{Z}_i^F[B]$, где $\overline{\alpha}_n=(\alpha_1,...,\alpha_n)$ – новые трансцендентные элементы из $\mathbb{A}$. Пусть $\overline{y}_k^{\overline{e}_k}$ являются корнями неприводимых над $\mathbb{Z}$ многочленов $\overline{e_k'}$. Так как $\mathbb{A}\supseteq\mathbb{Z}_\omega^F[B]$, то $A_{\overline{e_k}}$ – бесконечное множетсво, тогда кольцо $\mathbb{Z}[\overline{\beta}_m,\overline{\alpha}_n](\overline{y}_k^{\overline{e}_k})$ изоморфно вкладывается в некоторое $\mathbb{Z}[\overline{\beta}_i](\overline{x}_i^{\overline{f}_i})$. Теорема доказана.

Рассмотрим изоморфизмы структуры $\mathbb{Z}_\omega^F[B]$, оставляющие множество B трансцендентных элементов на месте.

ТЕОРЕМА 7. Число неизоморфных экзистенциально замкнутых компаньонов $\mathbb{Z}_\omega^F[B]$ кольца целых чисел $\mathbb{Z}$ континуально.

Доказательство. Пусть $\mathrm{E}=\{\varepsilon:\varepsilon=(a_1,a_2,...),\ a_2\in\{0,1\}\}$ – множество всех двоичных последовательностей. Положим $F_\varepsilon=(f_1,f_2,...)$, где

$$f_n(\beta_1,x_n)=\begin{cases}x_n^2-\beta_n, \text{если } \varepsilon=0,\\ x_n^2-\beta_n-1, \text{ если } \varepsilon=1.\end{cases} \qquad (2)$$

Тогда $\mathbb{Z}_\omega^{F_\varepsilon}[B]=\cup\mathbb{Z}_i^{F_\varepsilon}[B]$ – алгебраическое замыкание $\mathbb{Z}[B]$ неприводимыми многочленами f_i над $\mathbb{Z}$. Из теоремы 5 следует, что для $\varepsilon_1\neq\varepsilon_2$ кольца $\mathbb{Z}_\omega^{F_{\varepsilon_1}}[B]$ и $\mathbb{Z}_\omega^{F_{\varepsilon 2}}[B]$ неизоморфны, следовательно и их экзистенциальные расширения неизоморфны. Теорема доказана.

Заключение

Общая теория компаньон-классов Фрэсе и их теорий, разрабатываемых А.Т.Нуртазиным, составляет отдельную новую область в теории моделей. Этот подход, приложенный к конкретным классическим структурам и их теориям дает новые инструменты для исследований этих объектов. Изучение компаньон–класса кольца целых чисел обнаруживает области целостности, содержащие трансцендентые и, возможно алгебраические элементы со специальными свойствами многочленов определяющих эти элементы как компаньоны кольца целых чисел. Экзистенциально замкнутые компаньоны кольца целых чисел являются алгебраическими замыканиями согласованных между собой многочленов.

Благодарности. Настоящая работа выполнена в Институте Информационных и Вычислительных Технологий при поддержке Комитета Науки Министерства Образования и Науки Республики Казахстан (Грант №AP05135285 Теория индуктивных, экзистенциально замкнутых и форсинг-компаньонов и позитивно экзистенциально замкнутых моделей).

Литература

1 Barwise J. Robinson A., Completing theories by forcing // Ann. Math. Logic. – 1970. – №2. – P.119–142.

2 Robinson A. On the Metamathematics of Algebra.– Amsterdam.– North – Holland, 1951.– 540p.

3 Белеградек О.В. Алгебраически замкнутые группы // Алгебра и логика. – 1974. – Т. 13. – С. 239–255.

4 Cohen P.J. Set Theory and the Continuum Hypothesis. – NY., Benjamin, 1966.– 467 c.

5 Ершов Ю.Л., Палютин Е.А., Тайманов А.Д. Теория моделей.– Справочная книга по математической логике.– М.: "Наука", 1982, Ч. 1. – 492 с.

6 Fraisse R. Sur quelquesclassificationsdes systemesderelations // Publ.scient. de l'univ. d'Algers.– 1955. – A1. – P. 35–182.

7 Macintyre A. On algebraically closed groups // Ann. Math..– 1972. – Vol.96.–P. 53–97.

8 Macintyre A. Omitting quantifier-free types in generic structures // Journ. of Symbolic Logic.– 1972.– № 37.– P.512 - 520.

9 Nurtazin A.T. Countable infinite existentially closed models of universally axiomatizable theories// Siberian Advances in Mathematics.– 2016.– №26.–P. 99–125.

10 Нуртазин А.Т. Свойства экзистенциально замкнутых моделей // Алгебра и логика.– 2018.– Т.57, №3.– С. 321–327.

11 Nurtazin A.T. Properties of exictentionally cloused companions // Algebra and Logic.– 2018.– Vol.57, №3. – P. 211–221.

12 Нуртазин А.Т. Вынуждение формул в структурах и классах Фрэсе // Алгебра и логика.– 2018. – Т.57, №5.– С. 567–586.

13 Nurtazin A.T. Forcing formulas in Fraise structures and classes // Algebra and Logic.– 2018.– Vol. 57, №5. – P. 368–380.

14 Нуртазин А.Т. Счетные экзистенциально замкнутые модели универсально аксиоматизируемых теорий// Математические труды.– 2015.– Т.18, №1. – С. 48-97.

15 Нуртазин А.Т. Введение в теорию моделей, элиминация кванторов и экзистенциальная замкнутость. – Алматы: НЦ ГНТЭ, 2017.– 187 с.

16 Нуртазин А.Т. Любой форсинг-тип реализуется в некоторой форсинг-модели // Четырнадцатая междунар. азиатская школа-семинар «Проблемы оптимизации сложных систем».– Иссык-Куль, 2018.– С. 109–113.

17 Хисамиев З.Г., Алимжанова С.А. Компаньоны кольца целых чисел // Матер.научн.конф. ИИВТ МОН РК «Современные проблемы информатики и вычислительных технологий». – Алматы, 2019. – С.361–365.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ОБРАБОТКА ИНФОРМАЦИИ

УДК 004.42 10.5281/zenodo.3904110

С.Н. Верзунов, *verzunov@hotmail.com*
Институт машиноведения и автоматики НАН КР, Бишкек, Кыргызстан

РАЗРАБОТКА КРОССПЛАТФОРМЕННОГО ПРОГРАММНОГО КОМПОНЕНТА ТРАССОИСКАТЕЛЯ

В настоящей статье предложена архитектура кроссплатформенного программного компонента трассоискателя, основанная на разделении кода, зависимого от целевой платформы от кода, который может без каких-либо изменений запускаться на любой целевой платформе. Кроссплатформенность достигнута с помощью таких тулкитов как Kivy и инструментальных средств сборки приложений CMake, Swig и Buildozer, позволяющих портировать программный компонент трассоискателя на мобильную операционную систему Android, сохранив при этом его работоспособной и на десктопных операционных системах Windows и Linux. Это позволяет увеличить потенциальный круг возможных пользователей, и расширить функциональные возможности трассоискателя за счет применение встроенного во многие мобильные устройства gps-приемника и магнитного компаса, что также повышает удобство его практического использования.

Ключевые слова: кроссплатформенная архитектура, Kivy, buildozer, Android, Swig, ЛКАРД E502, ARM, python-for-android, Docker.

Введение

В работе [1] разработана архитектура программного компонента трассоискателя на базе устройства сбора данных ЛКАРД E502, основанная на использовании языка Python и модулей, предназначенных для обработки и визуализации данных. Как показали лабораторные испытания программной реализации данной архитектуры [2], важнейшим требованием, предъявляемым к такому программному обеспечению, является его кроссплатформенность, то есть способность работать на стационарных и портативных вычислительных устройствах с процессорами различной архитектуры под управлением разнообразных операционных систем. Это необходимо как в целях увеличения потенциального круга пользователей, так и для расширения функциональных возможностей трассоискателя, связанных с применение встроенного во многие мобильные устройства gps-приемника и магнитного компаса, а также удобства его практического использования. Трассоискатель с кроссплатформенным программным компонентом можно использовать на стационарных компьютерах с архитектурой процессора x86-64 для отладки и настройки аналоговой части трассоискателя, так и без изменения исходного кода на смартфонах и планшетных компьютерах с архитектурой процессора с архитектурой ARM под управлением операционных систем Android, iOS, и Windows 10 Mobile. В разработанном ранее варианте программного компонента кроссплатформен-

ность, по большей части уже была достигнута за счет применения языка программирования Python, и поэтому разработанный программный компонент может успешно работать под управлением операционных систем Windows и Linux, и, в частности возможна реализация трассоискателя в виде независимого устройства на базе одноплатного компьютера Raspberry Pi. Однако, некоторые части программного компонента трассоискателя, такие как графический интерфейс пользователя, библиотека визуализации данных matplotlib и драйвер платы сбора данных, тесно связаны с оборудованием и поэтому так или иначе должны быть портированы, т.е. адаптированы соответствующим образом, чтобы успешно взаимодействовать с различными устройствами используемой программно-аппаратной среды.

Постановка задачи

Задачей настоящего исследования является разработка и практическая реализация кроссплатформенной архитектуры программного компонента трассоискателя, выбор оптимальных методов и инструментальных средств обеспечения портирования программного компонента трассоискателя, позволяющих с минимальными усилиями, без изменения исходного кода использовать его на вычислительных устройствах с разнообразным аппаратным окружением управляемым различными операционными системами.

Материалы и методы

Kivy – это современное инструментальное средство (toolkit, тулкит) построения графического интерфейса пользователя, позволяющее создавать эргономичные интерфейсы для широкого спектра устройств. Скорость выполнения Kivy сопоставима с нативной мобильной альтернативой Java для Android или Objective C для iOS [3]. Основная идея, являющаяся ключевой для понимания всех преимуществ данного тулкита – это модульность и абстракция. Архитектура Kivy абстрагирована от таких базовых задач, как открытие окна, отображение графики и текста, воспроизведение звука, получение и отправка информации с устройств ввода вывода, и так далее. Это делает API (application programming interface, программный интерфейс приложения) расширяемым и простым в использовании. И, что важнее, это позволяет использовать низкоуровневые модули операционной системы, в которой запускается приложение. Например, в iOS, Andoroid и Windows существуют собственные API для подобного рода базовых задач.

В настоящее время Kivy активно развивается и имеет большое количество пользователей и разработчиков по целому ряду причин:

- Kivy имеет удобную в использовании встроенную поддержку мультитач-устройств с сенсорным экраном;
- Kivy специально создан для разработки кроссплатформенных графических приложений на языке Python и имеет более лучшую поддержку этого языка, по сравнению с адаптированными для него Qt и GTK +3;
- Kivy предоставляет более удобные API по сравнению с более ранними инструментами, такими как HTML и CSS) для построения графического интерфейса. Язык разметки интерфейса Kivy представляет собой адаптированный для использования совместно с Python вариант языка HTLM, отличающийся лаконичным, и поэтому более удобным синтаксисом.

- Kivy позволяет разрабатывать и поддерживать приложение без изменения исходного кода для множества операционных систем.

Последний пункт реализуется благодаря python-for-android [4] – инструменту для сборки с открытым исходным кодом, позволяющим упаковывать код на языке Python в отдельные APK (Android Package – формат архивных исполняемых файлов-приложений) для Android. Их можно передавать, устанавливать или загружать в магазин приложений Play Store, как и в любое другое приложение для Android. Этот инструмент изначально был разработан для кроссплатформенного графического тулкита Kivy, но в настоящее время поддерживает множество различных опций упаковки и может быть применен для создания любых приложений, написанных на языке Python.

Тулкит python-for-android реализует основные методы портирования приложения на операционную систему Android. В частности, он может скомпилировать интерпретатор Python, его зависимости для конкретной версии операционной системы, используемые в приложении библиотеки и код приложения Python для устройств с операционной системой Android. Этот этап является полностью настраиваемым, при этом самыми важным параметром является список зависимостей, т. е. модулей которые требуются приложению для работы. В случае программного компонента трассоискателя это модули для работы с ГИС (геоинформационной системой), вычислительными методами обработки и визуализации данных, подробнее о которых говорится в работе [1]. Аналогичный тулкит существует и для операционной системы iOS – Kivy iOS [5].

Для удобного управления всеми параметрами компиляции и сборки приложения под конкретную платформу используется такое инструментальное средство как Buildozer [6]. Это инструмент, позволяющий быстро упаковать мобильное приложение. Он автоматизирует весь процесс сборки, загружает необходимые компоненты, такие как python-for-android, Android SDK, NDK и т. д. В настоящее время Buildozer поддерживает упаковку для таких операционных систем как:

- Android: с помощью python-for-android (необходим компьютер с Linux или OSX, чтобы компилировать приложения для операционной системы Android);
- iOS: с помощью Kivy iOS. (для этого необходим компьютер с операционной системой OSX).

Поддержка других платформ включена в план (например, .exe для Windows, .dmg для OSX и т. д.) и будет реализована в ближайшее время. Процесс упаковки с помощью Buildozer управляется правилами, описанными в файле с именем buildozer.spec в каталоге приложения, определяя необходимые для его работы условия и такие параметры как заголовок, значок, используемые ресурсы (такие как, например, изображения, электронные подписи и д.р.) и библиотеки. Один и тот же файл спецификации используется для создания пакета для Android, iOS и т. д. Таким образом, реализуется возможность работы приложения на различных операционных системах без изменения исходного кода. В этом файле необходимо указать необходимые версии SDK, NDK, и другие зависимости приложения от среды, в которой оно будет работать. В частности, для портируемого программного компонента обязательно необходимо в качестве зависимостей указать модуль pyler [7], из которого импортируется набор функций, необходимых для работы со встроенным во многие устройства gps датчиком, который, в свою очередь, необходим для работы модуля ГИС mapview [8] – виджета Kivy для отображения ин-

терактивных карт. Кроме того импортируется и модуль graph, используемый в программном компоненте трассоискателя для отображения интерактивных графиков.

Другой важной частью, зависимой от аппаратного обеспечения частью является драйвер платы сбора данных Л КАРД E502. В целом он предоставляет собой три библиотеки:

- x502api – содержит общие функции для обоих модулей. Должна включаться в любой проект, работающий с одним из модулей, за исключением проектов, написанных только для L502 до появления библиотеки x502api, которые могут использовать только l502api;
- l502api – содержит специфические функции для модуля L502, а также функции, оставленные для совместимости с проектами, написанными до появления x502api;
- e502api – содержит специфические функции для модуля E502 [9].

Для портирования программного компонента необходимо адаптировать две библиотеки – x502api и e502api (т. к. в нем в настоящее время используется только модуль E502). Для ОС Windows предоставляется общий установщик «L-Card L502/E502 SDK», автоматически устанавливающий все необходимые драйвера, динамические библиотеки в системную директорию, а также все файлы, необходимые для подключения библиотеки к проекту приложения и примеры работы с ним в указанную директорию. Установка для ОС Linux так же не предоставляет особых усилий. Можно, например воспользоваться готовыми собранными пакетами, предоставляемыми «ЛКард». Это рекомендованный способ для дистрибутивов, для которых предоставляются собранные пакеты. Список поддерживаемых дистрибутивов приведен в документе [10].

Тем не менее, предоставляемые производителем исходные коды драйвера не являются полностью кроссплатформенными, так как ООО Л Кард не предоставляет поддержку операционных систем Android и iOS. Однако анализ исходных кодов показывает, что для них используется система сборки CMake. Эта система является свободным инструментом с открытым исходным кодом, основным разработчиком которого выступает компания Kitware. Название системы расшифровывается как «cross-platform make» (кроссплатформенная система сборки). Разработка инструмента ведётся с 1999 г., в качестве прототипа была использована утилита pcmaker, написанная в 1997 г. одним из авторов CMake. В настоящее время инструмент внедряется в процесс разработки многих программных продуктов, в качестве примеров широко известных проектов с открытым кодом можно привести KDE , MySQL , Blender , LLVM + clang и многие другие [11]. Принцип работы инструмента CMake напоминает принцип работы Buildozer: из каталога исходных кодов считывается файл CMakeLists.txt с описанием проекта, на выходе инструмент генерирует файлы проекта для одной из множества целевых операционных систем. Требования для сборки самого CMake включают наличие утилиты make и интерпретатора сценариев на языке bash либо скомпилированного инструмента CMake одной из предыдущих версий, а также компилятора C++ . При этом в исходных кодах CMake преднамеренно используются только возможности языка и стандартной библиотеки, поддерживаемые достаточно старыми версиями компиляторов. Таким образом, CMake является кроссплатформенным инструментом, переносим на большое количество платформ, что и позволяет модифицировать и скомпилировать драйвер устройства сбора данных Л КАРД E502 для мобильных операционных систем.

Для создания провайдера устройства ввода данных с платы Л КАРД Е 502 для библиотеки Kivy требуется SWIG – инструмент разработки программного обеспечения, который связывает программы, написанные на C и C ++, с различными языками программирования высокого уровня. SWIG используется с различными типами целевых языков, включая распространенные языки сценариев, такие как Javascript, Perl, PHP, Python, Tcl и Ruby. Список поддерживаемых языков также включает языки, не относящиеся к сценариям, такие как C#, D, Go, Java, включая Android, Lua, OCaml, Octave, Scilab и R. Также поддерживаются несколько интерпретируемых и скомпилированных реализаций Scheme (Guile, MzScheme / Racket) [12].

И наконец, для того, чтобы обеспечить повторяемую сборку программного компонента «Перспектива», результаты которой бы не зависели от используемой операционной системы и программного окружения, необходимо использовать инструментальное средство Docker. Контейнеры Docker предоставляют простые быстрые и надёжные методы разработки, распространения и запуска программного обеспечения, особенно в динамических и распределённых средах [13]. Docker позволяет создать контейнер, содержащий новейшие версии операционной системы и всех необходимых инструментов: Python, Kivy, Buildozer, CMake и др.

Таким образом, для портирования программного компонента необходимо решить целый ряд задач: разработать общую архитектуру, объединяющую графический интерфейс пользователя, драйвер платы сбора данных, тулкит Kivy и провайдер платы сбора данных в единую систему, отделив, однако, при этом платформозависимые и платформонезависимые части. Для реализации этой архитектуры потребуется доработать драйвер платы сбора данных и процесс его сборки, так, чтобы его можно было использовать на мобильных операционных системах, разработать архитектуру провайдера платы сбора данных и реализовать ее, обеспечив при этом повторяемую сборку платформозависимой части программного компонента трассоискателя для различных операционных систем.

Предлагаемое решение

Как уже было сказано выше, графический интерфейс пользователя, драйвер платы сбора данных и некоторые другие части программного компонента, как уже было сказано выше, тесно связаны с оборудованием и, в связи с этим, должны быть соответствующим образом адаптированы для обеспечения возможности его работы на операционных системах Android, iOS или Windows 10 Mobile. Схематически кроссплатформенная архитектура программного компонента, способного работать на мобильных операционных системах выглядит так, как показано на рис. 1. В предложенной архитектуре основной код программного компонента трассоискателя отделен от кода, зависящего от целевой рабочей платформы. Для портирования программного компонента требуется лишь собрать для конкретной операционной системы и типа процессора, показанную на рис. 1 платформозависимую часть. Тогда основной код программного компонента на языке Python можно будет запускать на любой платформе без изменений.

Сетевой стек TCP/IP, необходимый для связи с устройством сбора данных с помощью Ethernet или Wi-fi в настоящее имеется в любой современной мобильной и настольной операционной системе, однако не все функции поддерживаются одинаково хорошо.

Для того чтобы обеспечить работу на различных платформах в драйвере Л КАРД Е502 необходимо отключить поддержку автоматического поиска устройств в локальной сети, недоступную в настоящее время в операционной системе Android, изменив в файле *CMakeLists.txt* библиотеки e502api опцию

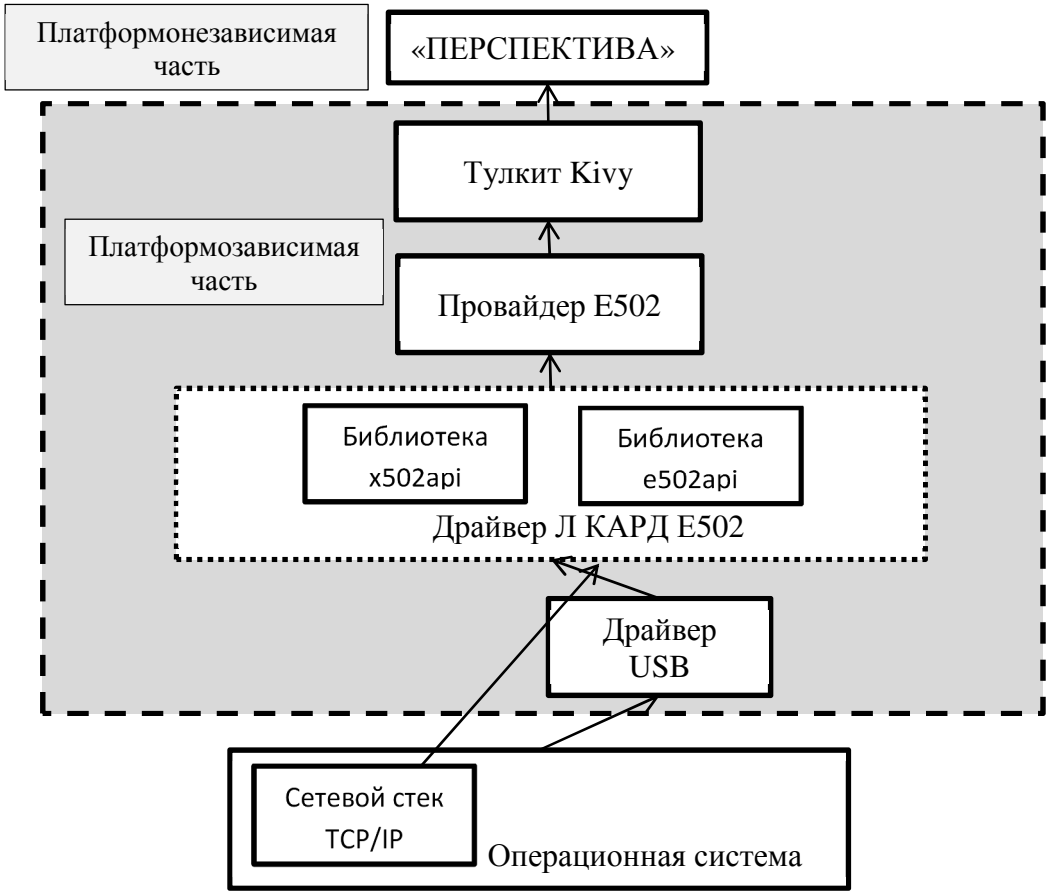


Рисунок 1 – Кроссплатформенная архитектура программного компонента трассоискателя на базе устройства сбора данных Л КАРД Е 502

option(E502API_ENABLE_DNSSD "enable dns-sd service discovery" OFF)

а в настройках программного компонента «Перспектива», в случае подключения устройства сбора данных по сети, необходимо указать IP адрес используемого устройства.

Кроссплатформенный драйвер USB можно найти по адресу [14], и добавить код:

```
if(android)
    include_directories(libusb-1.0 libusb-1.0/android)
…
if(linux)
    include_directories(libusb-1.0)
…
if(WIN32)
    include_directories(libusb-1.0 libusb-1.0/msvc)
…
```

в файл *CMakeLists.txt* библиотеки e502api. В настоящее время кроссплатформенным драйвером USB поддерживаются такие операционные системы как Linux, macOS, Windows, OpenBSD/NetBSD и Haiku.

В главный CMake-файл драйвера платы сбора данных для поддержки Android устройств следует добавить код:

```
if(${CMAKE_SYSTEM_NAME} MATCHES Android)
   set(CMAKE_SYSTEM_VERSION 21) # уровень API
   set(CMAKE_ANDROID_ARCH_ABI armeabi)
   set(CMAKE_ANDROID_STL_TYPE gnustl_static)
endif()
```

обеспечивающий корректную сборку для устройств на базе операционной системы Android. Как видно из приведенного выше кода, гарантируется работа на устройствах с API 21 и выше, что соответствует версии Android не менее чем 5.0. Работа на более старых устройствах, к сожалению невозможна, т.к. для них отсутствует поддержка стандарта POSIX Threads, необходимого для работы драйвера Л КАРД E502, реализующего поддержку многопоточных программ преимущественно ориентированных на исполнение на системах с общей памятью (Symmetric Multiprocessing, сокращённо SMP). Как известно, это такие системы, где установлено несколько процессоров или\и многоядерные процессоры и каждое ядро имеет доступ ко всей оперативной памяти компьютера [15].

Кроме того в файл /lib/osspec/osspec.c [16] необходимо внести исправление для компиляции драйвера платы сбора данных для операционной системы Android:

```
if (0) {/*timeout != OSSPEC_TIMEOUT_INFINITY*/
    struct timespec timeToWait;
    f_get_abs_time(timeout, &timeToWait);
    /*wt_res = pthread_timedjoin_np(thread, NULL, &timeToWait);*/
} else {
        wt_res = pthread_join(thread, NULL);
```

заменив тем самым неподдерживаемую в Android функцию стандарта POSIX Threads pthread_timedjoin_np на поддерживаемую во всех операционных системах функцию pthread_join.

На рис. 2 показана архитектура разработанного провайдера устройства ЛКард E502. Для начала работы с модулем необходимо установить с ним связь с помощью функции open_usb. Для идентификации устройств используется их серийные номера. Получить список серийных номеров всех подключенных устройств E502 можно с помощью get_usb_serial_list. Данная функция возвращает список, в котором сохранены найденные серийные номера, а принимает максимальное количество присоединенных модулей (по умолчанию это значение равно 16). Следует отметить, что с одним модулем одновременно может быть установлено только одно соединение. При попытке открыть модуль, с которым уже установлено соединение через другой описатель _hnd (возможно, в другой программе) open_usb вернет ошибку. При этом get_usb_serial_list по умолчанию возвращает список всех серийных номеров устройств, включая те, с которыми уже установлено соединение. Если нужно получить список только тех

устройств, с которыми еще не установлено соединения, то в get_usb_serial_list можно передать аргумент *only_not_opened=True*.

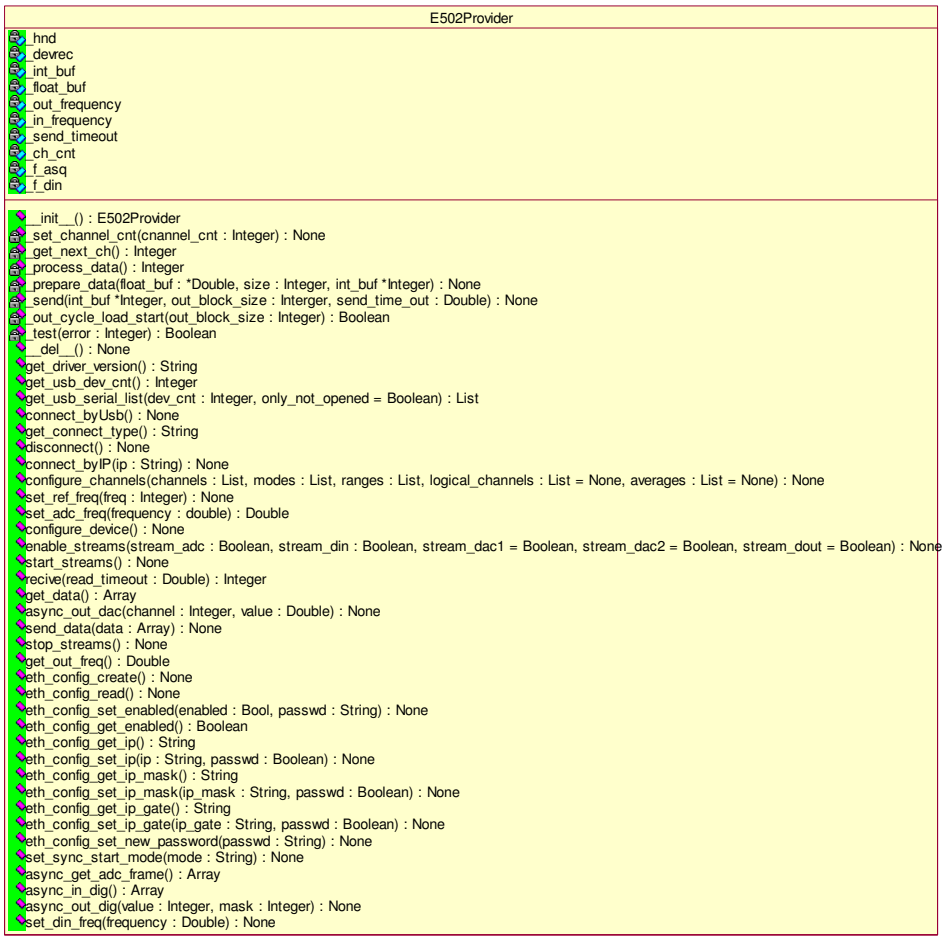


Рисунок 2 – Архитектура провайдера устройства ЛКард E502

Далее была разработана программа на языке Python, автоматизирующую создание контейнера Docker, и уставку в него операционной системы Ubuntu 18.04 и все вышеперечисленные инструментальных средств сборки. В этом контейнере был собран драйвер Л КАРД E502 и провайдер платы сбора данных Л КАРД E502 для библиотеки Kivy, состоящей из динамически разделяемой библиотеки и модуля для интерпретатора языка Python, созданных с помощью SWIG.

Наконец, был разработан файл buildozer.spec, требующийся для сборки APK-файла для операционной системы Android. В этом файле, кроме списка зависимостей и описания приложения, о которых уже говорилось выше, были указаны скомпилированные файлы провайдера E502; расширения файлов, содержащих необходимые ресурсы; минимальная версия Android API, необходимая для работы приложения; файлы драйвера Л КАРД E502 и архитектура процессора целевой платформы:

source.include_patterns = aarch64/python/*.so, aarch64/python/*.py
source.include_exts = py,kv,so
android.minapi = 21
android.add_libs_armeabi_v7a = aarch64/*.so, aarch64/devs/e502/*.so
android.arch = armeabi-v7a

garden_requirements = mapview, graph

Практическое исследование и выводы

Получившийся в результате сборки с помощью Buildozer APK-файл для проверки был установлен на планшет Google Nexus 7 3G (2012) и запущен (рис. 3) на

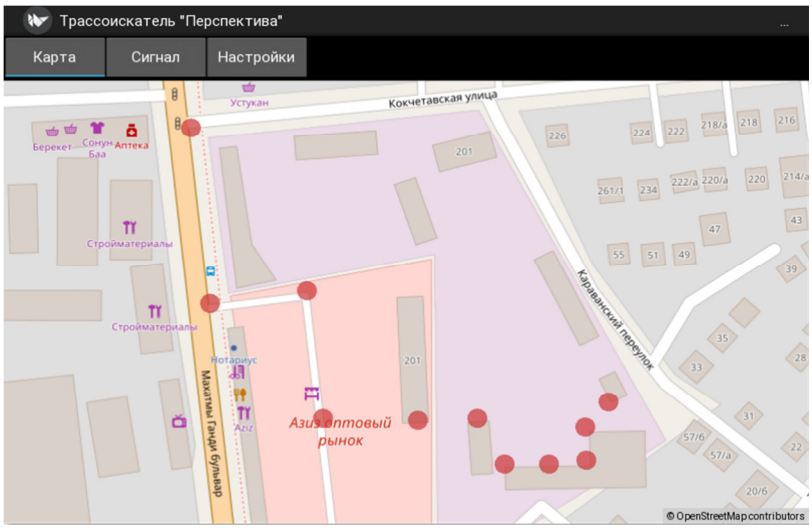


Рисунок 3 – Графический интерфейс кроссплатформенного программного компонента трассоискателя для работы с картой

операционной системе Android 7.2 с использованием ГИС OpenStreetMap [17]. Так же могут быть использованы и другие поставщики географических данных, например, Thunderforest или Google Maps. Портированный программный компонент трассоискателя позволяет наносить на карту найденные трассы кабелей на карту, выделяя их различными цветами, масштабировать карту до любых требуемых размеров и перемещать фокус ввода в любую необходимую для работы локацию.

Тестирование показало полную работоспособность портированного программного компонента трассоискателя и на смартфоне Meizu M2 Note с операционной системой Android 5.1. На рис. 4 показан графический интерфейс программного компонента трассоискателя для отображения принятых с помощью платы ЛКард E502 аналоговых и цифровых сигналов.

Заключение

Таким образом, в целях увеличения потенциального круга пользователей и для расширения функциональных возможностей трассоискателя, связанных с применением встроенного во многие мобильные устройства gps-приемника и магнитного компаса, а также удобства его практического использования была разработана и протестирована кроссплатформенная архитектура программного компонента трассоискателя и выбраны инструментальные средства ее реализации. Предложенная архитектура основана на разделении кода, зависимого от целевой платформы от кода, который может без каких-либо изменений запускаться на любой платформе. С помощью таких тулкитов как Kivy и инструментальных средств сборки кроссплатформенных приложений CMake и Buildozer удалось портировать программный компонент трассоискателя на мобильную операционную систему Android, сохранив при этом его работоспособной и на десктоп-

ных операционных системах Windows и Linux. В будущем возможно портирование и на другие мобильные устройства на базе операционных систем iOS и Windows 10 Mobile.

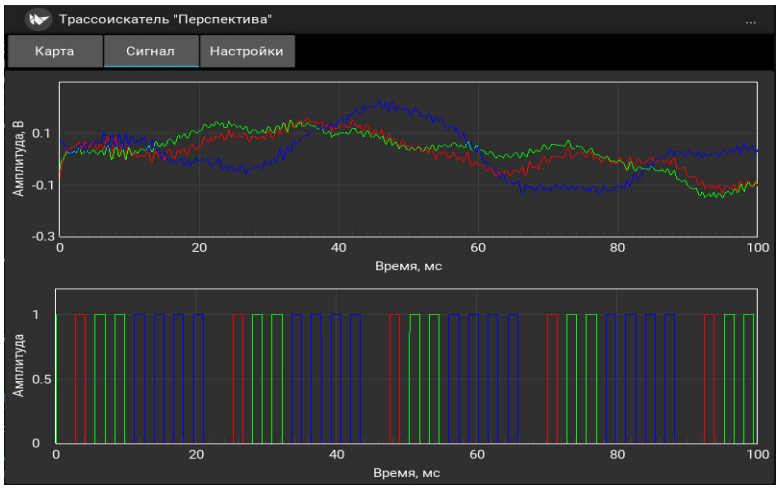


Рисунок 4 – Графический интерфейс кроссплатформенного программного компонента трассоискателя для отображения принятых сигналов.

Литература

1. Верзунов С.Н. Бочкарев И.В. Разработка программного компонента трассоискателя на базе устройства сбора данных Л КАРД E502 // Электротехнические системы и комплексы. 2018, №2(39). – С. 42-48.
2. Верзунов С. Н. Программный компонент трассоискателя на базе устройства сбора данных Л КАРД E502, ПК ПЕРСПЕКТИВА. Свидетельство об официальной регистрации программы для ЭВМ № 519 Кыргызская Республика, 27 августа 2018 г.
3. Dusty Phillips Creating Apps in Kivy – O'Reilly Media, 2014, 125 p.
4. https://python-for-android.readthedocs.io/en/latest/ (дата обращения 06.06.2019)
5. https://kivy.org/doc/stable/guide/packaging-ios.html (дата обращения 06.06.2019)
6. https://buildozer.readthedocs.io/en/latest/ (дата обращения 06.06.2019)
7. https://plyer.readthedocs.io/en/latest/ (дата обращения 06.06.2019)
8. https://mapview.readthedocs.io/en/latest/ (дата обращения 06.06.2019)
9. Борисов А. Современные устройства сбора данных L502/E502. Руководство программиста – М.: – ООО «Л Кард», 2016, 126 с.
10. Борисов А. Использование внешних репозиториев «L Card» для дистрибутивов Linux – М.: – ООО «Л Кард», 2019, 3 с.
11. Дубров Д. В. Система построения проектов CMake: учебник / Д. В. Дубров ; Южный федеральный университет. – Ростов-на-Дону: Издательство Южного федерального университета, 2015. – 419 с.
12. http://www.swig.org/doc.html (дата обращения 07.06.2019)
13. Моуэт Э. Использование Docker –М.: ДМК-Пресс, 2017. –354 с.
14. https://github.com/libusb/libusb (дата обращения 07.06.2019)
15. William Stallings. Operating Systems - Internals and Design Principles, 7th Edition. – Prentice Hall, 2011.
16. https://github.com/verzunov/e502-api-python (25.10.2019)
17. https://www.openstreetmap.org (дата обращения 07.06.2019)

УДК 681.322.067 10.5281/zenodo.3904117

С.В. Корякин*, аспирант, s.koryakin@aknet.kg*
Институт машиноведения и автоматики НАН КР

РАЗРАБОТКА КОНЦЕПЦИИ ПОСТРОЕНИЯ ПРОГРАММНО-АППАРАТНОГО ЯДРА УНИВЕРСАЛЬНОЙ СРЕДЫ ПРОЕКТИРОВАНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ ЗАЩИЩЕННОГО ИСПОЛНЕНИЯ

В статье рассматриваются вопросы построения программно-аппаратного ядра универсальной среды проектирования автоматизированных систем защищенного исполнения (АСЗИ) с использованием встроенных систем реального времени (СРВ). Универсальная среда проектирования предполагает полный цикл разработки, обеспечивая возможность моделирования отработки процессов, алгоритмов и проектируемых систем. Приведено описание, концепции построения разработанной универсальной среды проектирования АСЗИ. Предложена концепция построения и функциональная структура модели универсальной среды проектирования АСЗИ. Сформулирована задача построения универсальной СП. Принята классификация составных модулей СП. Обозначена открытость основных составных модулей СП, что позволит использовать новые модули, позиции, а также различные их комбинации основных составных модулей СП в ходе реализации возможных вариантов моделей в предложенной среде проектирования используемые для моделирования структур конкретных АСЗИ. Результаты от проведенных в статье исследований могут позволить стать концептуальной основой для проектирования функциональных блоков АСЗИ и проработки форм представления полученных результатов, составляющих модель универсальной среды проектирования АСЗИ. Полученный при работе с универсальной средой проектирования АСЗИ результат, позволит применять предложенную технологию модульного проектирования АСЗИ, что позволит значительно расширить возможности применения моделей универсальной среды проектирования и на ее основе повысить эффективность применения моделирования АСЗИ в современных условиях. Представленные наработки на основе предлагаемой концепции проектирования АСЗИ позволят повысить качество работы соответствующих организаций, а использование предложенных рекомендаций обеспечит существенное снижение рисков, возникающих при внедрении самой АСЗИ.

Ключевые слова*: модель; этапы проектирования; среда проектирования; системы реального времени; автоматизированная система защищенного исполнения.*

Введение

На современном этапе технологического развития наиболее значимой становится роль автоматизации систем управления в различных отраслях промышленности. На сегодняшний день внедрение подобных систем обеспечивает более качественное управление производством, сводя к минимуму участие человека в этих процессах и исключая тем самым ошибки, связанные с человеческим фактором. Развитие и разработка автоматизированных систем управления дает возможность улучшать многие направления деятельности: телекоммуникации, производство, экологию, экономику, энергетику, и другие.

В настоящее время во всем мире, в науке и других сферах деятельности, активно развивается направление по созданию инструментов для моделирования, проектирования, настройки и сдачи по ТУ встроенных СРВ управления различными процессами. С этой целью предлагается провести ряд работ по созданию универсальной среды проектирования (СП) автоматизированных систем защищенного исполнения (АСЗИ), работающих в режиме реального времени.

Универсальная среда проектирования АСЗИ должна поддерживать все технические циклы разработки создаваемых систем управления процессами (СУП) и необходимого для этого программного обеспечения. Тем самым обеспечивая качественную обработку всех вычислительных алгоритмов СУП и давая возможность скоростного имитационного моделирования этих систем. Применение СП позволяет обеспечить выпуск всей необходимой документации на создание и проектирование АСЗИ, такой как технологической, конструкторской, программной.

Разработка предлагаемой СП, позволит проводить эффективное моделирование встроенных СУП на этапе их проектирования, а также проводить анализ работы АСЗИ в режиме реального времени на основе полученных результатов от проводимых испытаний [1].

Главная задача, которая ставится при построении среды проектирования, – получить инструмент, с помощью которого станет возможным проводить работы по построению, пуско-наладке и запуску в эксплуатацию АСЗИ используя принцип моделирования объекта управления, как самой АСЗИ управления, так и всех взаимосвязей.

Следует отметить, что сам процесс создания АСЗИ управления объектами невозможен без моделирования процесса взаимодействия проектируемой системы и объекта управления. При этом должен соблюдаться принцип максимальной универсальности, а применение единых технологий на разных уровнях, в свою очередь, упростит задачу создания сложных многоуровневых систем комплексной автоматизации (СКА). Кроме того, распространение принципа универсальности на программное обеспечение позволяет сформулировать соответствующие основные требования к используемым программным продуктам.

Следует учитывать, что ряд устройств СКА (в особенности компьютеры) развивается настолько стремительно, что через полтора два года новейшая модель становится, в лучшем случае, моделью среднего уровня. То же самое, только с немного большими сроками, относится и к операционным системам. Наиболее консервативными элементами СКА из аппаратной части являются устройства ввода-вывода, а из программной – пакеты автоматизированного проектирования, а также расширения реального времени для операционных систем.

Кроме того, система обязательно должна позволять работать в режиме реального времени, производить измерения и передавать результаты измерений в дежурно-диспетчерский центр необходимых параметров.

Варианты концепции построения СП

Известно, что используемые операционные системы должны быть из числа наиболее применяемых в настоящее время, что фактически означает ориентацию на системы Windows, Linux, Free BSD.

Существующие различия между отдельными версиями этих систем имеют скорее «вкусовое», чем функциональное значение, а отдельные версии легко взаимодействуют между собой в рамках локальной сети и одинаково поддерживают различные программные пакеты. В связи с этим вполне допустимо иметь различные версии системы на различных рабочих станциях (модулях) среды проектирования. В этом случае достигается максимально свободная масштабируемость и простота реализации при модернизации среды проектирования.

Обобщая, можно констатировать, что при проектировании АСЗИ должны соблюдаться следующие принципы:

- Обеспечение ввода вывода сигналов через любые интерфейсы, с необходимыми характеристиками;

- Максимальная совместимость работы с другими системами и программным обеспечением;
- Максимальная эргономичность;
- Способность работать в режиме реального времени;
- реконфигурирование отдельных элементов комплекса и его структуры для получения различных вариантов;
- взаимодействия вычислительных средств;
- подключение дополнительных вычислительных средств (дополнительных узлов сети);
- различная коммутация модулей по физическим сигналам ввода-вывода с целью создания различных физических конфигураций;
- конфигурирование сигналов дискретного, аналогового и цифрового ввода-вывода с целью получения номенклатуры сигналов с требуемыми параметрами (дифференциальные сигналы и сигналы с общей точкой, различные уровни аналоговых, цифровых и дискретных сигналов и др.).

Традиционно при обсуждении концепции построения систем проектирования уделяется внимание особенности построения ядра АСЗИ: одноядерная или многоядерная. Большое внимание уделяется особенностям реализации, очень важным является реализация многозадачности процессов системы. По способу реализации задач системы проектирования делят на однозадачные и многозадачные. В большинстве случаев предпочтение отдается многозадачности. *Многозадачная ОС,* решает проблемы распределения ресурсов и в полной мере реализует мульти-программность, при этом характерны следующие особенности построения:

– Многозадачность, реализует идею разделения по времени, называется вытесняющим (preemptive). При этом каждой программе выделяется квант процессорного времени, по истечении которого управление передается другой программе. При этом первая программа будет вытеснена. В вытесняющем режиме работают пользовательские программы большинства коммерческих ОС.
– В большинстве ОС пользовательская программа способна монополизировать работу процессора, другими словами работает в режиме не вытеснения, не подлежит вытеснению код с ОС. Привилегированные программы, задачи реального времени, не вытесняются.
– Изучая детально различные ОС можно судить о приблизительности классификации. В ОС MS-DOS можно организовать запуск дочерних задач и наличие в памяти нескольких задач одновременно. Но эта ОС традиционно считается однозадачной, в основном из-за отсутствия защитных механизмов и возможностей к коммуникации.

Еще одним и немаловажным фактором при обсуждении концепции построения систем проектирования является поддержка многопользовательского режима работы системы. [3]

Кроме этого системы проектирования (СП) должны быть охарактеризованы допустимым временем реакции на внешнее событие, для запуска программы управления объектом. Система должна иметь запас ресурсов для поступающих данных от различных источников одновременно.

Основываясь на этом примем следующие основные критерии выбора СП:

1. Обеспечение эволюционности развития АСЗИ;
2. поддержка полного жизненного цикла АСЗИ;

Исходя из принятых критериев, полный жизненный цикл АСЗИ должен обеспечивать решение следующих задач (Таблица №1):

Таким образом, можно утверждать, что для существующих АСЗИ необходимо обеспечивать переход от существующей среды эксплуатации в вновь созданную с минимальными конструктивными изменениями и поддержкой всех ранее используемых в АСЗИ программ, исходных кодов, а также с поддержкой баз данных и приложений, до начала работ по обновлению системы. [4]

Применение указанных критериев возможно только после предварительного анализа средств проектирования АСЗИ. Это позволит максимально эффективно использовать имеющийся в арсенале парк программных и аппаратно-технических средств.

Таблица 1 – Основные задачи, обеспечивающие жизненный цикл АСЗИ

№	Наименование
1	Оценка стоимости, сроков разработки, прогнозирования и трудоемкости
2	Менеджмент разработки и сопровождение АСЗИ (планирование, координация и контроль за ресурсами и процессами)
3	Администрирование АСЗИ (оптимизация характеристик при эксплуатации)
4	Интеграция с существующими разработками (включая реинжиниринг приложений, конвертирование БД)
5	Внесение изменений в ТЗ и управление версиями ПО и конфигурацией АСЗИ
6	Пусконаладочные процессы и мероприятия
7	Адаптация к техническим платформам и СУБД
8	Разработка проектной документации с учетом требований проектных стандартов
9	Разработка распределенных баз данных (с выбором оптимальных вариантов распределения) [3]
10	Обобщенная, территориально распределенная разработка приложений с использованием всевозможных инструментов программ и средств (интеграция, тестирование и отладка)
11	Обследование формализованного описания предметной области к ее реальным моделям)
12	Декомпозиция и интеграция проекта на составные части
13	проектирование приложений и их моделей (логики приложений и пользовательских интерфейсов)
14	прототипирование приложений
15	проектирование баз данных

Анализ существующих средств проектирования АСЗИ

В настоящее время все существующие СП могут включать в своем составе следующие основные категории:

1. CASE-системы – независимые (upper CASE) и "клиент-серверные" средства разработки интегрированные в СУБД. В качестве примера можно назвать:

- Westmount I-CASE+Uniface
- Designer/2000+Developer/2000

- Другие

Главное преимущество этих систем заключается в том, что при использовании становится возможным полностью разрабатывать АСЗИ. При этом учитываются все существующие функциональные спецификации, логические процессы, пользовательские интерфейсы, БД и многое другое. При этом проектируемые АСЗИ остаются в одной технологической среде. Кроме того инструменты 1 категории обладают сложностью, широкой сферой применения и высокой гибкостью к модификации.

2. Средства проектирования БД. Реализуют так называемую методологию "сущность-связь" ("entity-relationship") и рассматриваемые в предложенной среде проектирования средствами разработки приложений и прикладных программ. Примерами служат:

- SILVERRUN+JAM
- ERwin/ERX+PowerBuilder
- Прочие

Кроме этого средства проектирования можно классифицировать по следующим признакам:

a) степень интегрированности – локальные средства, частично интегрированных средства проектирования и обслуживания, охватывающие практически все этапы жизненного цикла АСЗИ, средства интегрированные на 100%, связанные репозиториями;
b) методологи и модели АСЗИ и БД;
c) % интегрированности в СУБД;
d) % открытости;
e) доступности площадок и платформ .

В число доступных и популярных в использовании средств проектирования попадают и дешевые системы с ограниченными функциональными возможностями, и дорогостоящие продакшин системы для использования в различных платформах и операционных средах. В настоящее время, современный рынок ПО насчитывает большое количество различных CASE-систем, которые широко используются во всем мире.

Благодаря стремительному развитию науки и техники от пользователей применяющих средства проектирования требуются знания. Мировой опыт показывает, что использование средств проектирования требует длительного времени, но по мере приобретения практических навыков в использовании и общей культуры проектирования АСЗИ эффективность применения средств проектирования будет резко возрастать, при этом наибольшая потребность в использовании специализированных средств проектирования АСЗИ, необходимо на начальных этапах разработки. А именно на этапе анализа и составлении спецификаций и требований. Все это обусловлено тем, что цена ошибок, допущенных на начальных этапах разработки, в геометрической прогрессии повышает цену ошибок, выявленных на поздних этапах разработки АСЗИ.

В настоящее время, рынок ПО располагает средствами проектирования которые по популярности использования занимают лидирующее место это[3]:

- Westmount I-CASE;
- Uniface;
- Designer/2000+Developer/2000 (ORACLE);
- SILVERRUN+JAM;
- ERwin/ERX+PowerBuilder.

Перечисленные средства проектирования не охватывают весь перечень существующих средств проектирования, но характеризуют их как наиболее популярные среди проектировщиков АСЗИ. Нельзя забывать о том, что на рынке ПО постоянно появляются все новые и новые системы, а так же новые версии и модификации перечисленных систем (например, CASE/4/0, System Architect и т.д.).

В качестве примера можно привести результаты анализа основных средств проектирования, приведенные ниже в таблице характеристик (Таблица 2).[3]

Анализ показывает, что наиболее полно удовлетворяет всем критериям средств проектирования АСЗИ, комплекс Westmount I-CASE+Uniface. К примеру в комплексе Westmount I-CASE+Uniface целостность базы проектных данных и единая технология сквозного проектирования АСЗИ обеспечивается за счет использования интерфейса Westmount-Uniface Bridge. Следует отметить, что каждый из двух продуктов сам по себе является одним из наиболее мощных в своем классе.[3]

Таблица 2 – Характеристики СП

СП	West-mount I-CASE + Uniface	Designer/2000+Developer/2000	SILVER-RUN + JAM	ERwin/ERX + PowerBuilder
Поддержка полного жизненного цикла ИС	+	+	+	+
Обеспечение целостности проекта	+	+	-	-
Независимость от платформы	+ (ORACLE, Informix, Sybase, Ingres и др., dbf-файлы)	- (целевая СУБД – только ORACLE)	+ (ORACLE, Informix, Sybase, Ingres и др.)	+ (ORACLE, Informix, Sybase, поддержка ODBC)
Одновременная групповая разработка БД и приложений	+	- *)	- *)	- *)

Таким образом, самыми адаптивными и практичными средствами разработки больших АСЗИ на сегодняшний день является, по мнению специалистов в этой области, комплекс Westmount I-CASE+Uniface. В случае использования не исключается возможность использования в том же самом проекте других средств, PowerBuilder, для разработки сравнительно небольших прикладных систем в среде MS Windows.[3]

Произведенный анализ говорит о том, что на сегодняшний день ни одно доступное средство проектирования не удовлетворяет всем основным критериям и не способно перекрывать все потребности необходимые при реализации проекта. Таким образом, появляется острая необходимость разработать и использовать универсальное средство проектирования, которое позволит построить единую технологическую среду. [3] Кроме того остается открытым вопрос защищенности, который практически не охвачен или охвачен стандартными средствами защиты которые не могут удовлетворять современным требованиям защищенности систем существующими средствами проектирования. Вопрос защищенности является наиболее важным для современных автоматизированных систем и может охватывать как аппаратную, так и программную защиту элементов системы, и несомненно требует отдельного рассмотрения. Поэтому, вопросу защищенности будет посвящена отдельная работа, в которой будут рассмотрены существующие и предложены альтернативные способы защиты автоматизированных систем.

Концепции построения ядра универсальной СП АСЗИ

Весь процесс проектирования можно условно разбить на 5 этапов (рис. 1):

1 – На первом этапе разработки, при проектировании системы, используются чисто математические модели. (В качестве основных аппаратно-программных технических средств используются обычные ПК (ЭВМ)).

2 – Используются две отдельные модели, обменивающиеся физическими сигналами объекта управления и системы управления. В качестве основных аппаратно – программных технических средств используются специализированные сервера (модули) в которых вычислительные возможности должны сочетаться с возможностями ввода-вывода используемых сигналов [2].

3 – Используется спроектированная и построенная модель объекта, для первоначальной её отладки и настройки.

4 – Поиск наиболее сложных нестыковок. Используется модель системы управления вместо самой системы.

5 – Ввод в эксплуатацию готовой к работе системы управления.

Таким образом, при разработке программно-аппаратного ядра универсальной АСЗИ (рис. 1), должны быть разработаны модули 1 и 2, кроме этого необходимо обеспечить различные варианты комбинирования при работе. Так, как после проведения анализа различных систем и средств проектирования выявлено следующие:

1. Все АСЗИ при разработке проходят указанные на рис. 1 этапы проектирования.

2. Отдельные элементы разрабатываемой среды проектирования использовались при проектировании и построении АСЗИ.

3. Все элементы при помощи которых происходит проектирование АСЗИ существуют в разрозненном состоянии и не всегда реализованы с использованием современных подходов.

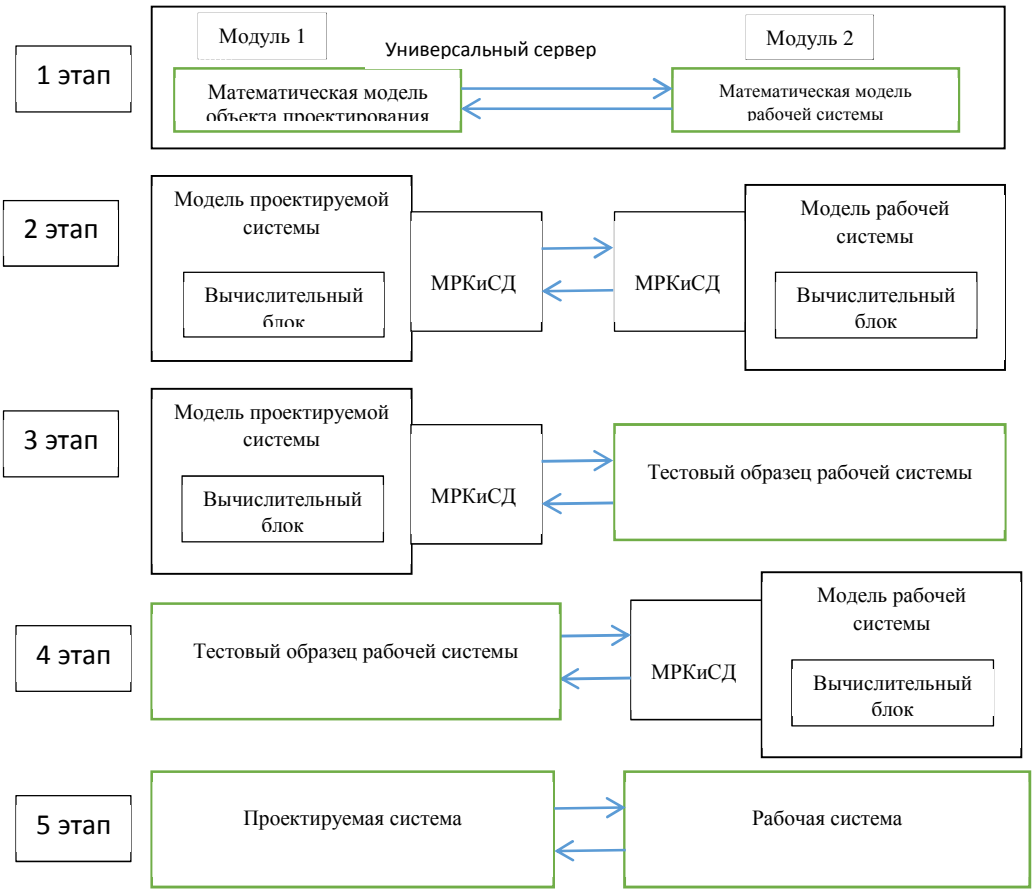


Рисунок 1 – Структурная схема процесса разработки вариантов моделей АСЗИ.

Это приводит:

1. К увеличению сроков и затрат на разработку,
2. Не позволяет осуществлять моделирование в необходимом объёме и с требуемым качеством.

С точки зрения функционирования, разрабатываемые модули 1 и 2 (рис. 1) должны будут представлять собой вычислительные комплексы, выполняющие следующие работы:

- Обработка математических моделей АСЗИ и решение различных уравнений;
- визуальное отображение полученных результатов (таблицы, графики, изображения объектов и др.);
- ввод-вывод физических сигналов различных уровней для обработки и анализа.

Разрабатываемую среду проектирования предполагается строить для решения задач моделирования с преследованием двух основных целей: уменьшение времени затраченного на строительство АСЗИ или СРВ и повышения качества. Соответственно, при этом должны быть созданы некоторые демонстрационные задачи, чтобы продемонстрировать работу комплекса, и контрольные задачи для его приёмки.

С учетом вышеизложенного предлагается использовать следующие концепции к формированию технологических средств ядра системы проектирования (рис. 2.): 1 – однослойная; 2 – многослойная; 3 – аппаратное ядро с наложенным на него программным слоем; 4 – набор прикладных программ с использованием цельного программного ядра; 5 – использование секционного программного ядра в виде прикладных программ.

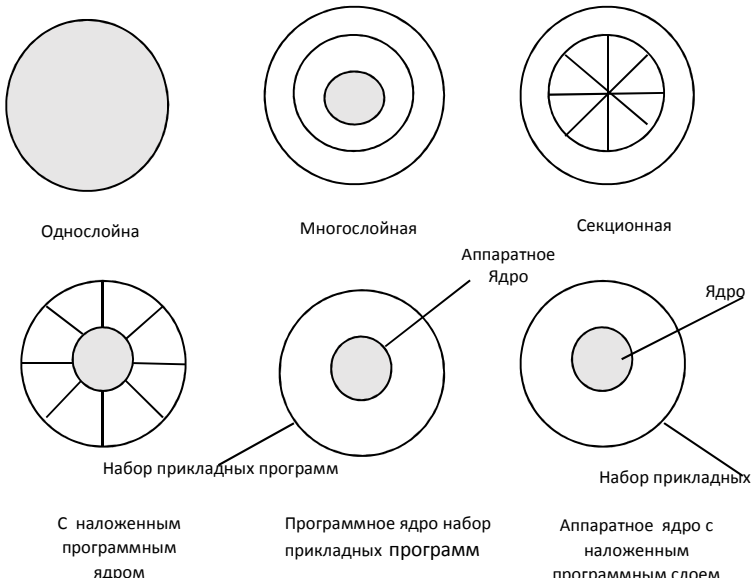


Рисунок 2 – Концепции построения ядра системы проектирования

Каждая из предложенных концепций построения ядра системы (рис. 2.), обладает рядом преимуществ и недостатков. В связи с этим предлагается создать такую среду проектирования, где возможно будет синтезировать все из предложенных вариантов и путем исключения слабых звеньев создавать наиболее оптимальную и универсальную систему, отвечающую всем необходимым требованиям и условиям, а также проводить апробации перед непосредственным внедрением спроектированной системы. Иными словами, предлагается использовать блочную архитектуру построения автоматизированных систем.

Говоря об АСЗИ в целом можно утверждать что, АСЗИ защищена лишь настолько, насколько защищено ее самое слабое звено. Зачастую слабым звеном

оказывается человек: проектировщик, создающий плохое решение под напором сложности; администратор, неправильно настраивающий систему; бизнесмен, предпочитающий предложить новые функции в ущерб надежности, или техник службы сопровождения, ставший жертвой мошенников, использующих «социальную инженерию», представляющую собой фактически совокупность методов взлома систем с использованием человеческой психологии.

В процессе разработки АСЗИ должен соблюдаться разумный компромисс между созданием встроенных неразделимых механизмов проектирования и блочных унифицированных средств и процедур проектирования и построения автоматизированных систем.

Только на этапе разработки автоматизированных систем можно полностью учесть взаимное влияние блоков и устройств автоматизированной системы, добиться системности защиты, надежности, и устойчивости.

Таким образом, предложенный принцип блочного построения автоматизированных систем реализует следующие преимущества:

- Минимизируется трудоемкость при разработке, отладке, контроле и верификация устройств, прикладных программ, алгоритмов АСЗИ;
- Используется параллельность при разработки блоков АСЗИ;
- Используются стандартные блоки и модули;
- Упрощается процесс модернизации АСЗИ;
- Появляется удобство и простата эксплуатации.

Указанные принципы блочной архитектуры автоматизированной системы, позволяют создать структуру приближенную к идеальной СП, которая эффективно сможет решать поставленные задачи и оптимизировать структуру АСЗИ таким образом, чтобы она была максимально эффективна в режиме реального времени и при необходимости могут быть использованы любые другие блоки системы, чтобы не допустить снижения эффективности применения по прямому назначению [6].

Стандартные входные и выходные интерфейсы блоков позволят упростить процесс модернизации системы, а также использовать аппаратные или программные блоки в качестве альтернативы, так же как это происходит при использовании семиуровневой моделью OSI и технологий и протоколов передачи данных.

При разработке автоматизированной системы любой сложности, необходимо предусматривать возможность ее развития в двух направлениях:

1. Увеличения количества пользователей;
2. Наращивание потенциала различных сетевых и других ресурсов АСЗИ по мере совершенствования и развития информационных технологий.

Таким образом, использование предложенных принципов позволит создать уникальное программно-аппаратное ядро универсальной среды проектирования автоматизированных систем защищенного исполнения.

Заключение

На основе проведенного анализа существующих программных сред проектирования АСЗИ предложен новый подход к решению проблем возникающих при построении, адаптации после внедрения и модернизации спроектированных моделей в составе универсальной среды проектирования АСЗИ работающих в режиме реального времени.

Суть предложенного подхода заключается в следующем:

1. Предложена концепция построения и функциональная структура модели универсальной среды проектирования АСЗИ.
2. Сформулирована задача построения универсальной СП.

3. Принята классификация составных модулей СП.
4. Обозначена открытость основных составных модулей СП, что позволит использовать новые модули, позиции, а также различные их комбинации основных составных модулей СП в ходе реализации возможных вариантов моделей в предложенной среде проектирования используемые для моделирования структур конкретных АСЗИ [7, 9].
5. Результаты от проведенных в статье исследований могут позволить стать концептуальной основой для проектирования функциональных блоков АСЗИ и проработки форм представления полученных результатов, составляющих модель универсальной среды проектирования АСЗИ.
6. Полученный при работе с универсальной средой проектирования АСЗИ результат, позволит применять предложенную технологию модульного проектирования АСЗИ, что позволит значительно расширить возможности применения моделей универсальной среды проектирования и на ее основе повысить эффективность применения моделирования АСЗИ в современных условиях.

Представленные наработки на основе предлагаемой концепции проектирования АСЗИ позволят повысить качество работы соответствующих организаций, а использование предложенных рекомендаций обеспечит существенное снижение рисков, возникающих при внедрении самой АСЗИ [8, 10].

Литература

1. *Курзыкина А.В.* Проблемы внедрения автоматизированной информационной системы // Молодой ученый. – 2017. – №4. – С. 124–167. – URL https://moluch.ru/archive/138/38806/ (дата обращения: 16.03.2020).
2. *ГомаХ. UML.* Проектирование систем реального времени, параллельных и распределенных приложений: Пер. с англ. – М.: ДМК Пресс, 2011. – 704 с. (Серия «Объектно-ориентированные технологии в программировании»). ISBN 9785-94074–723–9. – С. 115–174.
3. http://citforum.ru/database/kbd96/42.shtml, Обзор средств проектирования информационных систем А.М.Вендров, Центральный банк РФ. – С. 5–9.
4. *Вылегжанин О.Н., Шкатова Г.И.* Сравнительная оценка двух методов выбора наилучших линейных регрессоров // Применение математических методов и ЭВМ в медико-биологических исследованиях: Межвузовский научно-технический сборник. – Томск: Изд.во ТПУ, 1988. – С. 18–22.
5. *Брякин И.В.* Проектирование автоматизированных систем защищенного исполнения: научно-методическое пособие. Бишкек 2017. 24 с.
6. *Погребной В.К.* Системы реального времени. Моделирование и автоматизированное проектирование. – Томск: Изд. во ТПУ, 2006. – 209 с.
7. *Корякин С.В.* Современные тенденции развития систем информационной безопасности // Проблемы автоматики и управления. – 2017. – № 2 (33). – С. 82 – 85.
8. *Брякин И.В., Корякин С.В.* Информационная безопасность в системах реального времени // Проблемы автоматики и управления. – 2017. – № 2 (33). – С. 115 –118.
9. *Нестеров А.Л.* Проектирование АСУТП: методическое пособие. Кн.1. М: Деан, 2006. С. 552–553.
10. *Брякин И.В.* Методология итерационного проектирования магнитовариационной ИИС // Проблемы автоматики и управления. – 2011. –№1(20). С.20–30.

Кыргыз Республикасынын Улуттук илимдер академиясынын Машина таануу жана автоматика институту

КОНЦЕПЦИЯНЫ ИШТЕП ЧЫГУУ, ТҮЗҮҮ ПРОГРАММАЛЫК-АППАРАТТЫК ӨЗӨК УНИВЕРСАЛДУУ ЧӨЙРӨНҮ ДОЛБООРЛООНУН АВТОМАТТАШТЫРЫЛГАН СИСТЕМАЛАРДЫ КОРГОЛГОН АТКАРУУ / *С.В. Корякин*

Беренесинде маселелер каралат куруу программалык-аппараттык өзөк универсалдуу чөйрөнү долбоорлоонун автоматташтырылган системаларды корголгон аткаруу(АСЗИ) пайдалануу менен бириктирилип системасынын айкын убакыт ыргагында(СРВ). Чөйрө түзүлгөн системаларды тез моделдөө жана алгоритмдерди иштетүү мүмкүндүгүн камсыз кылуу менен иштеп чыгуунун толук циклин колдойт. Инвестиция агенттиги тарабынан иштелип чыккан универсалдуу долбоорлоо чөйрөсүн куруунун сыпаттамасы, концепциясы келтирилген.

Документтин түрү:
Модели; долбоорлоо этаптары; долбоорлоо чөйрөсү; реалдуу убакыт системасы (РУС); корголгон аткаруунун автоматташтырылган системасы.

post- graduate student
Institute of machine science and automation NAS KR

DEVELOPMENT OF THE CONCEPT OF BUILDING THE HARDWARE AND SOFTWARE CORE OF THE UNIVERSAL DESIGN ENVIRONMENT FOR AUTOMATED SYSTEMS OF PROTECTED EXECUTION // *S. V. Koryakin,*

The article deals with the issues of building the hardware and software core of the universal design environment for automated systems of protected execution (ASSI) using embedded real -time systems (SRS). The environment supports a full development cycle, providing high- speed simulation of the created systems and processing algorithms. The description of the concept of building the developed universal design environment of the ASSI is given.

Keywords: Model; design stages; design environment; real-time systems (SRV); automated system of secure execution.

УДК 004.89 10.5281/zenodo.3904130

Н.М. Лыченко, А.В. Сороковая
Институт машиноведения и автоматики НАН КР, Бишкек, Кыргызстан
E-mail: nlychenko@mail.ru, nastusha24sh-g@yandex.com

ПРИМЕНЕНИЕ LSTM-НЕЙРОННЫХ СЕТЕЙ ДЛЯ КЛАССИФИКАЦИИ ИНДЕКСА КАЧЕСТВА ВОЗДУХА Г. БИШКЕК

Рассмотрена задача прогнозирования индекса качества воздуха AQI г. Бишкек в зависимости от метеопараметров как задача нейросетевой классификации. Обоснован выбор LSTM- нейронной сети как наиболее эффективной. Разработан классификатор индекса качества воздуха, решающий проблему прогноза классов AQI "Хороший"/"Нездоровый", для различной истории наблюдений метеопараметров и различной глубины прогноза. Достигнута точность прогноза более 90%.

***Ключевые слова**:* классификация, прогноз, индекс качества воздуха, LSTM-нейронная сеть.

Введение. Одним из интегрированных показателей загрязненности атмосферного воздуха является индекс качества воздуха (Air Quality Index, AQI [1]. Индекс качества воздуха является кусочно-линейной функцией концентрации загрязняющих веществ в атмосфере: диоксида серы (SO_2), диоксида азота NO_2, взвешенных частиц меньше 10 мкм (PM_{10}) и взвешенных частиц меньше 2,5 мкм ($PM_{2.5}$), окиси углерода (CO) и озона (O_3). Для каждого из этих загрязнителей Агентство США по охране окружающей среды (EPA) установило национальные стандарты качества воздуха. В основе вычисления AQI – соотношение измеренной усредненной концентрации загрязнителя и стандартной (допустимой) концентрации этого загрязнителя. Значение AQI, равное 100, в целом соответствует национальному стандарту качества воздуха для загрязнителя. Поскольку AQI диоксида азота, диоксида серы и моноксида углерода чаще всего ниже 50, наибольшую угрозу для здоровья человека представляют озон и частицы PM. В общем, существует 6 уровней (классов) AQI:

- «Хороший» показатель AQI – от 0 до 50; качество воздуха считается удовлетворительным, а загрязнение воздуха представляет небольшой или нулевой риск;
- «Умеренный» AQI - от 51 до 100; качество воздуха приемлемое, однако, у очень небольшого числа людей могут возникнуть умеренные проблемы со здоровьем (например, люди, которые необычно чувствительны к озону, могут испытывать респираторные симптомы);
- «Нездоровый для чувствительных групп» AQI составляет от 101 до 150; люди с заболеванием легких и сердца, пожилые люди и дети подвергаются большему риску.
- «Нездоровый» показатель AQI - от 151 до 200; каждый человек может начать испытывать некоторые негативные последствия для здоровья, а члены чувствительных групп могут испытывать более серьезные последствия;
- «Очень нездоровый» AQI - от 201 до 300; такой уровень предупреждает о том, что каждый человек может испытывать серьезные последствия для здоровья;

- «Опасный» AQI больше 300; такой уровень соответствует чрезвычайным ситуациям, все население, скорее всего, будет затронуто негативным воздействием на здоровье.

Охрана окружающей среды предполагает не только оценку состояния здоровья населения в данный момент времени, но и прогнозирование последствий влияния загрязняющих веществ на здоровье жителей исследуемого региона. Поэтому задача анализа динамики изменения концентраций вредных веществ и построения моделей для прогноза их содержания в воздухе представляет особый интерес.

Для г. Бишкек проблема загрязнения атмосферного воздуха стоит крайне остро и построение прогностических моделей на основе актуальных для города данных – важная задача. В день, когда прогнозируется повышение AQI из-за загрязнения мелкими частицами, агентство по чрезвычайным ситуациям или организации общественного здравоохранения могут [1]:

- рекомендовать чувствительным группам, таким как пожилые люди, дети и лица с респираторными или сердечно-сосудистыми проблемами, избегать физических нагрузок на открытом воздухе;
- объявить «день действий» для поощрения добровольных мер по сокращению выбросов в атмосферу, таких как использование общественного транспорта;
- рекомендовать использовать маски для предотвращения попадания мелких частиц в легкие.

Во время периода очень низкого качества воздуха, когда AQI указывает, что острое воздействие может нанести значительный вред общественному здоровью, правительственные организации могут объявить о чрезвычайной ситуации и, согласно плану действий, ограничить выбросы основных источников загрязнения воздуха до тех пор, пока показатели степени загрязнения не опустятся до приемлемых значений.

Универсальных моделей для прогноза ИКВ быть не может, поскольку в них необходимо учитывать региональные природные, экономические, антропогенные и климатические особенности территории. В литературе опубликовано немало работ, связанных с построением моделей прогноза индекса качества воздуха для различных регионов и городов. Для г. Бишкек таких работ практически нет, главным образом, ввиду того, что открытая информация об уровне загрязненности атмосферного воздуха в городе появилась лишь в феврале 2019 г. на сайте [2]. Основываясь на этой информации, в [3] оценена временная изменчивость индекса качества воздуха и разработаны интегрированные модели авторегрессии-скользящего среднего (AutoRegressive Integrated Moving Average model, ARIMA-модели) для его краткосрочного прогноза на основе данных наблюдений для летнего периода 2019 года. В [4] выполнена оценка влияния метеорологических факторов (таких, как скорость ветра, температура, относительная влажность воздуха, температура точки росы, интенсивность осадков и атмосферное давление) на процесс загрязнения воздуха г. Бишкек частицами PM2.5 в период с февраля по ноябрь 2019 г. и выявлены умеренные корреляции (как положительные, так и отрицательные) между концентрациями PM2.5 и метеорологическими параметрами, измеренными в текущий и прошлые сроки. В связи с этим, интересной представляется задача прогноза индекса качества воздуха г. Бишкек с учетом метеорологических факторов.

Задача прогноза AQI как задача классификации. В настоящей работе задача прогноза индекса качества воздуха AQI по метеорологическим данным рассмотрена как задача классификации AQI в зависимости от метеорологических условий, соответствующих определенным уровням AQI.

Задача классификации в общем смысле представляет собой задачу разделения множества объектов на классы на основе обучающей выборки – конечного множества объектов, для которых классы заранее определены экспертом или иным способом [5]. Задача классификации заключается в построении алгоритма для определения классовой принадлежности объектов вне обучающей выборки.

Математически задачу классификации можно сформулировать следующим образом. Дано множество описаний объектов X и множество номеров (наименований) классов Y. Описание объекта $x \in X$ представляет собой вектор признаков $x = (f_1(x), f_2(x), \dots, f_n(x))$, называемый признаковым описанием объекта x. Существует неизвестное отображение $y' : X \to Y$, значения которого определены на обучающей выборке. Требуется построить алгоритм $a : X \to Y$, способный классифицировать любой объект $x \in X$.

Для AQI классы соответствуют определенным уровням загрязнения воздуха [1]. При анализе данных наблюдений AQI [2] в период с 6 февраля 2019 по 31 марта 2020 года было подсчитано количество наблюдений, указывающих на определенный класс AQI и их процентное соотношение (рисунок 1а)).

Из анализа количества наблюдений видно, что в подавляющем числе зафиксированы значения AQI класса “Умеренный”. Количества наблюдений для классов “Хороший” , “Нездоровый для чувствительных групп”, “Нездоровый”, “Очень нездоровый”, “Опасный” недостаточно для решения задачи классификации AQI по всем классам. В связи с этим будем классифицировать AQI по двум классам: “Хороший” (объединяет классы “Хороший” и “Умеренный”) и “Нездоровый” (объединяет классы “Нездоровый для чувствительных групп”, “Нездоровый”, “Очень нездоровый” и “Опасный”) (рисунок 1б)). При таком распределении данных, если всегда предсказывать “Умеренный” класс, то точность такого «наивного» прогноза составит 70%.

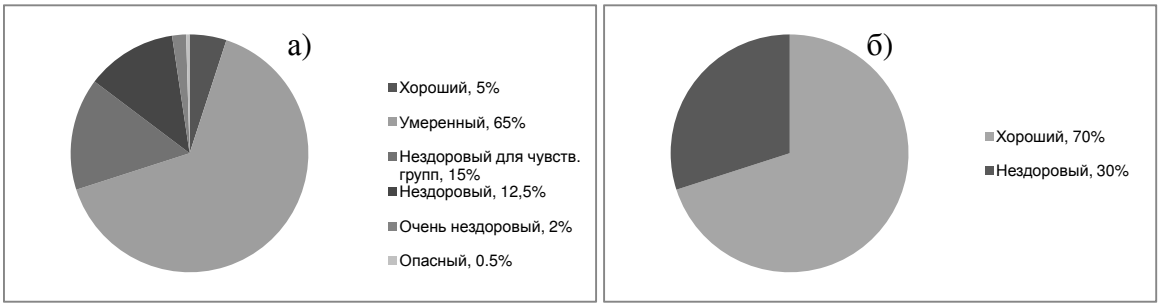


Рисунок 1 – Распределение наблюдений по классам AQI (а) и распределение наблюдений по объединенным классам AQI (б)

Выбор модели решения. Рассмотрим задачу классификации AQI на два класса в зависимости от метеорологических условий, соответствующих определенным уровням AQI. При этом будем считать классификатор приемлемым, если его точность превышает точность 70%.

Как показано в [3], на степень загрязнения воздуха влияют метеорологические условия не только на текущий момент, но и история изменения этих условий. Это говорит об инерционности процессов загрязнения, т.е. изменение происходит не моментально, а меняется постепенно вместе с изменением сопутствующих факторов. Очевидно предположить, что прогностические модели, учитывающие историю наблюдений, будут работать лучше, чем модели, основанные на оценке исключительно текущих данных. Для анализа исторических данных, как правило, используются искусственные нейронные сети, в частности, рекуррентные нейронные сети [6, 7].

Однако степень загрязнения воздуха может реагировать на различные факторы с разной скоростью. История каждого фактора должна быть оценена по-разному – для каких-то факторов важны только последние данные, влияние других может сказываться продолжительное время. Учитывая этот факт, в данной работе предложено для классификации AQI использовать LSTM-сети [8, 9]. Их архитектура содержит так называемые фильтры, которые в процессе обучения настраиваются сохранять/забывать информацию выборочно о различных факторах и, таким образом, могут взвесить влияние каждого фактора во времени.

LSTM-сеть (долгая краткосрочная память) – частный вид рекуррентной нейронной сети. Однако обычные рекуррентные сети могут учитывать только недавние прошлые состояния сети. Эту проблему называют проблемой долговременных зависимостей или проблемой исчезновения градиента [8]. В теории эта проблема решаема правильным выбором параметров сети, однако проблема выбора этих параметров все еще остается нерешенной. LSTM-сети созданы специально для решения этой проблемы и позволяют обнаруживать как длинные, так и короткие шаблоны в данных, а также частично устраняют проблему исчезновения градиента [8, 9].

Рекуррентная нейронная сеть может быть представлена в форме последовательности одинаковых модулей нейронной сети (рисунок 2) [6].

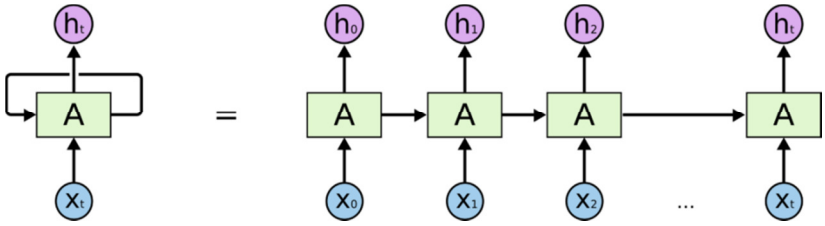


Рисунок 2 - Рекуррентная нейронная сеть в виде последовательности модулей

В обычной рекуррентной сети, как рассматривалось ранее, модуль представляет собой один слой нейронов с обычной функцией активации. Модуль LSTM-сети представляет собой не один, а четыре слоя, которые взаимодействуют особым образом (рисунок 3). Желтым прямоугольником соответственно обозначен слой нейронной сети, розовым кружком – поточечная операция, стрелками – поток передачи целого вектора, сходящимися стрелками – конкатенация векторов.

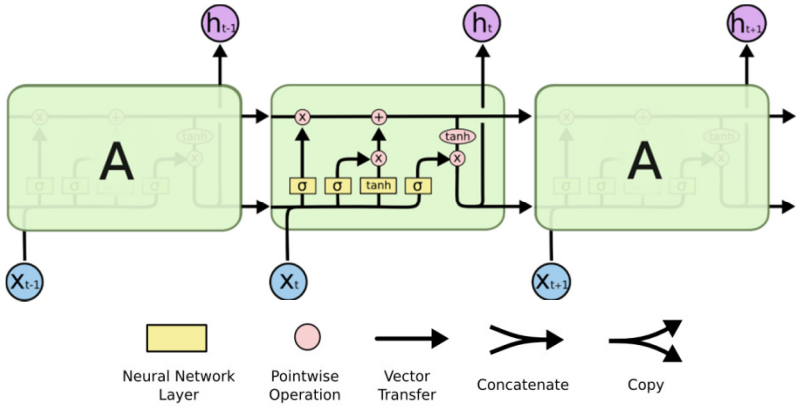


Рисунок 3 – Общая архитектура LSTM-сети

Ключевой компонент модуля LSTM-сети – вектор состояния (рисунок 4). Вектор состояния регулируется специальными фильтрами, они управляют удалением и обновлением информации в нем. Сигмоидальный слой в фильтре определяет, какую

долю информации перезаписать в векторе состояния (0 – не записывать ничего, 1 – записать все).

Условные обозначения здесь:

- x_t— входной вектор в момент времени t,
- h_t— выходной вектор в момент времени t,
- C_t— вектор состояний в момент времени t,
- W_k — матрица параметров слоя k, т.е. веса связей,
- b_k — вектор смещений выходов слоя k,
- f_t — вектор фильтра забывания в момент времени t,
- i_t — вектор входного фильтра в момент времени t,
- o_t — вектор выходного фильтра в момент времени t.

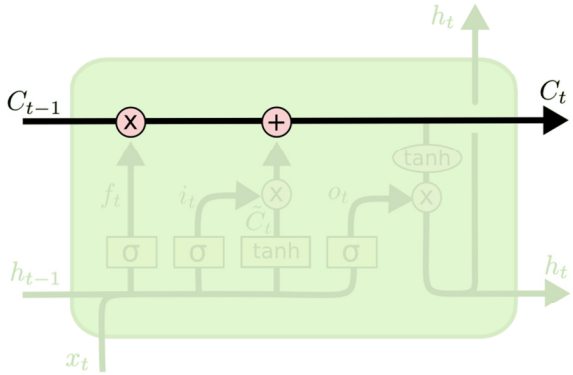


Рисунок 4 – Состояние модуля LSTM-сети

LSTM работает следующим образом [6]:

1. Слой фильтра забывания определяет, какую информацию можно удалить из вектора состояния (Рисунок 5a). Для каждого числа из состояния C_{t-1} возвращается число от 0 до 1 (где нулевое значение значит, что значение надо забыть).

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

2. Сигмоидальный слой входного фильтра определяет, какие значения следует обновить, а тангенс-слой строит вектор новых значений $\tilde{C}_t$, которые могут быть добавлены в состояние C_t (Рисунок 5b).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_C)$$

3. Для обновления состояния старое состояние необходимо умножить на f_t и прибавить к нему $i_t * \tilde{C}_t$ (Рисунок 5c).

4. Выход LSTM-сети представляет собой состояние C_t с примененной к нему функцией активации, при этом выходной фильтр определяет, какие именно элементы состояния выводить (Рисунок 5d).

$$h_t = o_t * tanh(C_t)$$
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

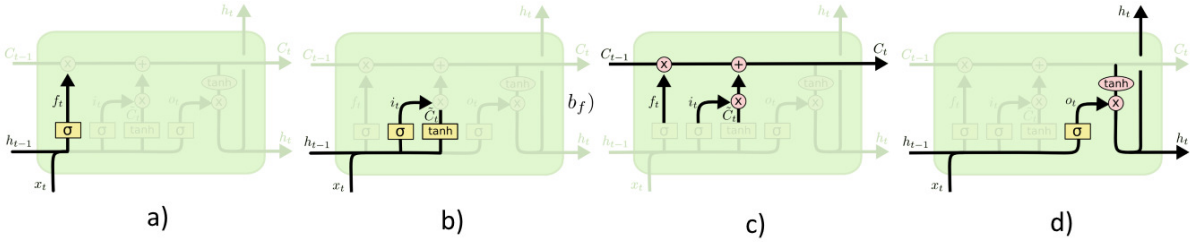


Рисунок 5 – Принцип работы LSTM-сети

Методы оценки качества классификации. Для проверки качества классификации необходима тестовая выборка – размеченный набор данных, т.е. набор объектов с номером (наименованием) класса, к которому они относятся. Классифицируя объекты тестовой выборки и сравнивая полученные классы с действительными, можно оценить качество классификации в виде численной метрики.

Простейшей метрикой является аккуратность (accuracy) – доля правильно классифицированных объектов из всей тестовой выборки [10]:

$$acc = \frac{P}{N}$$

где P – количество правильно классифицированных объектов, N – размер тестовой выборки. Эта метрика присваивает одинаковый вес всем объектам, что неправильно, если распределение классов в выборке различно.

В общем виде все метрики можно вывести из так называемой матрицы ошибок (confusion matrix). Матрица ошибок A представляет собой матрицу размера $n * n$, где n – количество классов, представленных в выборке. Элемент матрицы A_{ij} содержит значение, показывающее, сколько раз классификатор определил класс *j* как класс *i*.

Имея матрицу ошибок, можно вывести две метрики [10]:

- точность в пределах класса – доля объектов, действительно принадлежащих классу относительно числа объектов, которые классификатор определил к этому классу:

$$precision_c = \frac{A_{c,c}}{\sum_{i=1}^{n} A_{c,i}}$$

- полнота в пределах класса – доля объектов, определенных классификатором к классу, относительно всех документов, действительно принадлежащих этому классу:

$$recall_c = \frac{A_{c,c}}{\sum_{i=1}^{n} A_{i,c}}$$

Для получения единой метрики значения точности и полноты для каждого класса обычно усредняются.

Метрикой, объединяющей точность и полноту, является так называемая F-мера – гармоническое среднее между точностью и полнотой [10]:

$$F = \frac{2 * precision * recall}{precision + recall}$$

Таким образом, чем больше F-мера, тем качественнее работает классификатор.

Подготовка данных. Для решения задачи классификации были подготовлены два файла данных за период наблюдений с 06.02.2019 по 31.03.20: с историческими наблюдениями AQI [2] и историческими наблюдениями метеорологических условий [11], с интервалом наблюдений в 3 часа.

На рисунке 6 представлены график изменения индекса качества воздуха AQI г. Бишкек за указанный период и графики автокорреляционной (ACF AQI) и частичной автокорреляционной (PACF AQI) функций. Из графика автокорреляционной функции видно, что в данных содержится суточная периодичность в 24 часа (8 лагов).

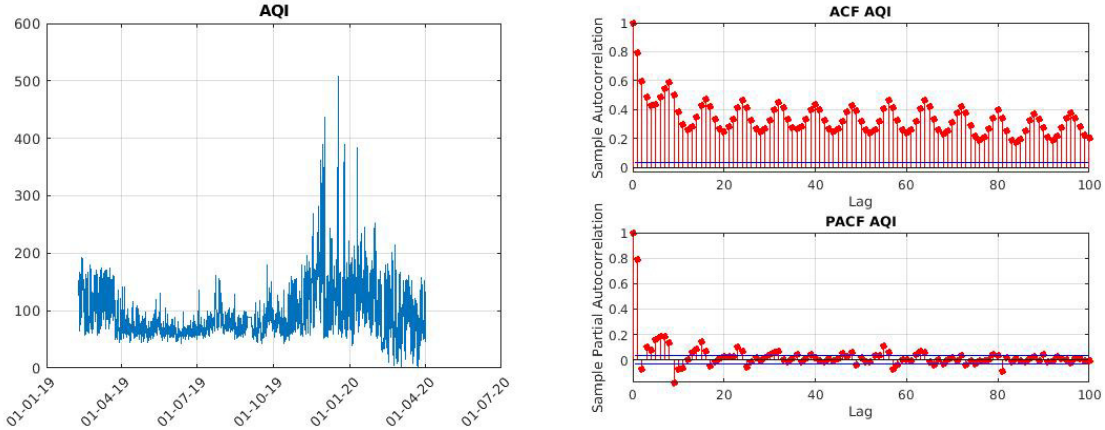


Рисунок 6 – Значения индекса качества воздуха в г. Бишкек за период 06.02.2019-31.03.2020, автокорреляционная (ACF AQI) и частичная автокорреляционная (PACF AQI) функции.

Входной вектор классификатора определяется пятью параметрами: температура воздуха, атмосферное давление, относительная влажность, скорость ветра, температура точки росы. При этом данные о температуре воздуха, атмосферном давлении, температуре точки росы нормализуются с помощью *Z*-нормы.

Выходной вектор, к которому должен приближаться выход классификатора, определяется двумя параметрами – вероятностью отнесения выхода классификатора к классу “Хороший”, равной 1 для AQI <= 100 и вероятностью отнесения выхода классификатора к классу “Нездоровый”, равной 1 для AQI > 100.

При проведении вычислительных экспериментов по прогнозированию класса AQI варьировались следующие параметры:

- *S* – длина последовательности векторов исторических данных - входных векторов (ВВ) классификатора;
- *P* – глубина прогноза (на сколько шагов вперед прогнозируется AQI). Шаг прогноза – 3 часа.

Программная реализация проведения экспериментов извлекает данные, соотнося их по времени, нормализует данные по заданной функции нормализации, по заданным *S* и *P* генерирует все возможные последовательности входных векторов классификатора длиной *S* и сопоставляет им значения AQI в моменты времени, удаленные на *P* шагов от момента последних значений этих последовательностей.

Так как анализ показал несбалансированность данных (70% - относятся к классу AQI “Хороший” и 30% - к классу “Нездоровый”), для обучения классификатора необходимо было сбалансировать эти группы. С этой целью из выборки “Хороший” случайным образом выбрано такое же количество примеров, которое содержится в выборке “Нездоровый”.

Разработка классификатора. Для подтверждения эффективности LSTM-сети в решении задачи классификации AQI проведены эксперименты на различных однослойных сетях с одинаковым количеством нейронов на скрытом слое (50 нейронов с тангенциальной функцией активации). Эксперимент проводился на выборке наблюдений с 06.02.2019 по 26.11.2019.

Тестируемые модели классификаторов – следующие.

1. Полносвязная сеть прямого распространения, не учитывающая исторические данные. Вход – вектор признаков в один момент времени.
2. Полносвязная сеть прямого распространения, учитывающая исторические данные. Вход – вектор, конкатенирующий векторы входной последовательности.
3. Рекуррентная нейронная сеть. Вход – последовательность векторов.
4. LSTM-сеть.

Выход каждой сети – вероятности того, что AQI в момент времени через P шагов от момента, соответствующего последнему вектору последовательности, будет отнесен к классу “Хороший” и “Нездоровый”. Предсказанный класс определяется большей вероятностью.

Для каждой модели проведено 3 эксперимента. В каждом эксперименте из набора данных выделяются 5 непересекающихся частей. Согласно алгоритму перекрестной проверки, каждая из этих частей единожды выступает в качестве тестовой выборки, а оставшиеся части – в качестве обучающей. F-мера точности классификатора в каждом эксперименте определяется средним значением F-меры точности, полученной на каждом этапе перекрестной проверки. Результаты каждого эксперимента также усреднялись. В таблице 1 представлены полученные значения F-меры точности перечисленных однослойных моделей.

Из таблицы видно, что предположение о том, что классификатор, учитывающий историю изменения векторов, должен давать более высокую точность, чем учитывающий только последние данные, действительно верно. При этом LSTM-сеть действительно оказалась наиболее эффективной из рассмотренных.

Таблица 1 – Оценка F-меры точности моделей – различных однослойных сетей

Номер тестируемой модели	Эксперименты (F-мера точности)			Среднее
	I	II	III	
1	0.7340	0.7279	0.7282	0.7300
2	0.7630	0.7690	0.7800	0.7707
3	0.7969	0.8201	0.7898	0.8023
4	0.8214	0.8340	0.8368	0.8307

Дальнейшие эксперименты проводились на LSTM-сетях для выборки с 06.02.2019 по 31.03.2020.

Выбранная модель LSTM-сети несколько усложнена для предотвращения переобучения и возможного увеличения точности.

Структура используемой далее LSTM-сети:

1. Входной слой длиной 5 (5 признаков входного вектора BB), который принимает последовательность векторов признаков длиной *S*.
2. Первый скрытый слой – слой прямого распространения с числом нейронов 100 и тангенциальной функцией активации, отображает входные векторы в векторы большей длины, для дальнейшего внесения шума в данные.
3. Второй скрытый слой – слой регуляризации, который меняет некоторый процент значений выхода предыдущего слоя для предотвращения переобучения (вносит шумы);
4. Третий скрытый слой – LSTM-сеть с 50 нейронными модулями и тангенциальной функцией активации как основной классифицирующий слой;
5. Четвертый скрытый слой – слой прямого распространения с числом нейронов 10 и тангенциальной функцией активации;
6. Выходной слой – слой с 2 нейронами и функцией активации *SOFTMAX*. Функция взвешивает входы и предсказывает вероятности активации каждого нейрона. При этом сумма выходов нейронов всегда равна 1.

Все слои сети полносвязные, т.е. каждый нейрон имеет связь с каждым предыдущим нейроном, а для рекуррентных слоев (LSTM-сеть) каждый вход слоя также связан с каждым выходом слоя.

В качестве визуализации процесса обучения на рисунке 7 представлены графики изменения точности (Accurancy) и функции потерь (Loss) в зависимости от номера эпохи обучения (epoch) сети с параметрами S=12, P=8 для обучающей (train) и тестовой (validation) выборок. Заметим, что процесс обучения вполне отвечает рекомендациям, приведенным в [12].

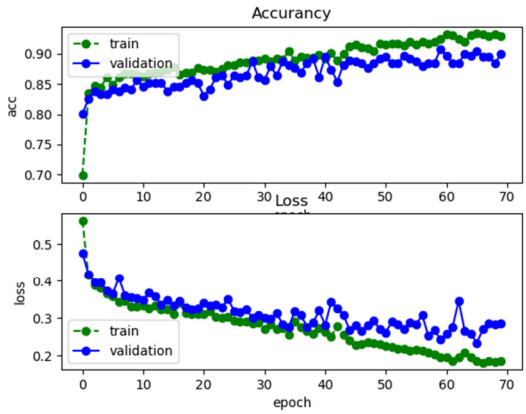


Рисунок 7 – Графики изменения точности (Accurancy) и функции потерь (Loss) в зависимости от номера эпохи обучения (epoch) сети с параметрами S=12, P=8.

Обсуждение результатов. Эксперименты с LSTM-сетью дали результаты, представленные в таблице 2.

Из экспериментов видно, что все классификаторы дали приемлемую точность прогноза (>70%), однако лучшую точность прогнозирования дали классификаторы, учитывающие историю данных глубиной 8-16 шагов (1 – 2 дня).

Таблица 2 – F-мера точности моделей в зависимости от длины последовательности входных векторов (*S*) и глубины прогноза (*P*).

S	P	F-мера
2	2	0.8842
2	4	0.8696
2	8	0.8617
2	16	0.8408
2	24	0.8615
2	32	0.8387

S	P	F-мера
4	2	0.8748
4	4	0.8787
4	8	0.8753
4	16	0.8684
4	24	0.8585
4	32	0.8552

S	P	F-мера
8	2	0.8976
8	4	0.8924
8	8	0.9011
8	16	0.8955
8	24	0.8889
8	32	0.8824

S	P	F-мера
12	2	0.9018
12	4	0.9021
12	8	0.9093
12	16	0.8972
12	24	0.8958
12	32	0.8998

S	P	F-мера
16	2	0.9120
16	4	0.8885
16	8	0.9085
16	16	0.9051
16	24	0.9083
16	32	0.9115

Как видно, модели, основанные на истории наблюдения погодных условий более 1 суток (S >= 8) мало отличаются по точности. Это можно связать с тем, что погодные условия, наблюдаемые за 24 часа, несут максимум полезной информации. Влияние более ранних погодных условий практически угасает.

При прогнозе AQI до 2х дней вперед (P <= 16) точность модели практически не меняется на исторических данных за 24 часа и более, F-мера остается примерно 90%. Таким образом в этом случае нецелесообразно для классификации AQI рассматривать историю изменения погодных условия длинее 24 часов.

При прогнозе на более чем 2 суток вперед (P > 16) более эффективными оказались модели, учитываюшие исторические данные в промежутке более 24 часов. Это влияние небольшое - F-мера увеличивается на 1–2%.

Таким образом, настоящим исследованием доказана эффективность рекуррентных сетей для классификации индекса качества воздуха. Представлен классификатор на основе LSTM-сети, решающий проблему прогноза классов AQI “Хороший”/“Нездоровый”. Найдены параметры классификатора, дающие достаточно хорошую точность прогноза. На основе полученных результатов можно сделать вывод, что прогноз AQI возможен до 4х дней вперед с точностью 88-90%, что значительно выше точности «наивного» прогноза (70%).

Проблемой при исследовании классификаторов стала недостаточность данных. Для обучения сети обычно требуется значительное количество примеров, в то время как при исследовании использовалась всего по 970 примеров каждого класса, учитывая удаление части выборки, соответствующей классу “Хороший” для сбалансированности данных. Увеличение истории наблюдений позволит увеличить количество примеров и построить классификатор AQI на большее количество классов. С этой же целью можно искусственно увеличивать количество примеров для «плохих» данных различными методами (oversampling).

Литература

1. Air Quality Index (AQI) – A Guide to Air Quality and Your Health. US EPA. 9 December 2011.
2. AirNow Depatment of State // https://airnow.gov/index.cfm?action=airnow.global_summary #U.S._Department_of_State $Bishkek, (дата обращения: 30.04.2020).
3. Верзунов С.Н., Лыченко Н.М. Краткосрочное прогнозирование индекса качества воздуха на основе ARIMA-моделей // Математическое и компьютерное моделирование: сборник материалов VII Международной научной конференции (22 ноября 2019г.). – Омск: Изд-во Омск. гос. ун-т, 2019.
4. Лыченко Н.М. Регрессионный анализ метеорологических факторов и концентраций частиц PM2.5 в атмосферном воздухе г. Бишкек // Проблемы автоматики и управления. – 2019. №2 (37). – С. 5-15.
5. Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И. Методы и модели анализа данных: OLAP и Data Mining. – СПб.: БХВ-Петербург, 2004. – 336 с.: ил.
6. X. Zhao, R. Zhang, J.-L. Wu, P.-C. Chang. A Deep Recurrent Neural Network for Air Quality Classification // Journal of Information Hiding and Multimedia Signal Processing. – V.9, N.2, March 2018.
7. B. Carremans. Forecasting Air Pollution with Recurrent Neural Networks. – Nov 19, 2018 https://towardsdatascience.com/forecasting-air-pollution-with-recurrent-neural-networks-ffb095763a5c (дата обращения: 30.04.2020).
8. Deep Learning, NLP, and Representations Posted. – URL: http://colah.github.io/posts/2014-07-NLP-RNNs-Representations/ (дата обращения 06.07.2019).
9. Understanding LSTM Networks. – URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (дата обращения 14.09.2019).
10. Оценка классификатора (точность, полнота, F-мера). – URL: http://bazhenov.me/blog/2012/07/21/classification-performance-evaluation.html (дата обращения 29.05.2019)
11. Сайт «Расписание погоды rp5.ru» Архив погоды в Бишкеке https://rp5.ru/%D0%90%D1%80%D1%85%D0%B8%D0%B2_%D0%BF%D0%BE%D0%B3%D0%BE%D0%B4%D1%8B_%D0%B2_%D0%91%D0%B8%D1%88%D0%BA%D0%B5%D0%BA%D0%B5 (дата обращения: 30.04.2020).
12. Курс CS231n: Convolutional Neural Networks for Visual Recognition. – URL: https://cs231n.github.io/neural-networks-3/#baby (дата обращения 01.09.2020).

УДК 004.65 10.5281/zenodo.3904132

Мустафин Р. Ш.*, *Осмонов М.С.
Кыргызско-Российский Славянский университет, г. Бишкек, Кыргызстан
Email: r.mustafin.work@gmail.com; msosmonov@gmail.com

АВТОМАТИЗАЦИЯ ПРОВЕРКИ РЕШЕНИЯ ЗАДАНИЯ ПО SQL

Способность формулировать корректные SQL-запросы является фундаментальным навыком, который требуется многим специалистам по разработке программного обеспечения.
Овладение этим навыком является сложным процессом, требующим от обучаемого значительных усилий и практики. Автоматизация процесса проверки навыков по правильному написанию запросов значительно повышает эффективность такого обучения. Несмотря на ряд предложенных решений, на настоящий момент отсутствуют единые подходы к решению задачи построения систем автоматизированного контроля знаний, а также требования к их программной реализации.
В данной статье предлагается подход к автоматизации проверки результата запросов SQL, базирующийся на использовании DML (Data Manipulation Language) инструкций, с возможностью проверки решения одного и того же задания на разных реализациях СУБД.

Ключевые слова: SQL, электронное обучение, решение заданий, СУБД, база данных, автоматизация.

Введение

Построение запросов к базам данных на языке SQL является ключевым навыком, который требуется многим разработчикам программного обеспечения, так как он лежит в основе практически любого программного обеспечения и используется для манипулирования данными и получения информации.

Несмотря на кажущуюся простоту (обычно для получения нужного результата необходим один или несколько операторов SQL, которые заменяют достаточно большой фрагмент кода на обычном языке программирования) научиться писать SQL-запросы – довольно сложная задача.

Процесс определения, какая информация требуется из базы данных в соответствующий оператор SQL является не таким простым, как кажется на первый взгляд, так как программное обеспечение базы данных выполняет множество операций, которые невидимы программисту.

Это особенно сложно, поскольку не виден результат, который будет при выполнении запроса возвращен из базы данных.

Обычно, для проверки правильности формирования SQL-запросов программисты создают запрос и проверяют возвращаемый результат. Если результат не соответствует желаемому, то запрос уточняется, и шаги проверки и уточнения повторяются, пока не будут получены результаты, удовлетворяющие требованиям. И эта рутинная работа выполняется вручную и визуально, что отнимает очень много времени и сил.

В данной статье рассматриваются вопросы автоматизации процесса обучения языку SQL, что позволяет упростить обучение студентов путем автоматизации проверок задач, выполняемых ими.

В первой части статьи приведено описание возможных формулировок заданий по SQL. Во-второй – рассматривается алгоритм автоматизации процесса проверки задания по SQL.

I. Описание задачи проверки решения задания по SQL

В обычном случае задача по SQL содержит следующие артефакты:

1) база данных (БД);
2) некоторое количество таблиц со связями или без внутри этой БД, которые могут содержать данные;
3) пользовательские процедуры или функции, ограничения, триггеры, индексы и т.д.
4) артефакты, специфичные для конкретной реализации СУБД [3].

Допустим, что задания сформулированы следующим образом:

- создать таблицы в БД с полями X, которые связаны между собой некоторым образом. Заполнить таблицы данными;
- изменить существующую схему данных (таблицы, связи, ограничения), чтобы соответствовать новым требованиям.
- написать запрос (или несколько запросов), который выдаст данные, удовлетворяющие требованиям по заданию**.**

Эти задания можно разделить на два класса: задачи на использование **DDL** (Data Definition Language) инструкций и задачи на использование **DML** (Data Manipulation Language) инструкций. Эти два класса требуют разные типы проверок.

DDL [4] инструкции используются для внесения изменений в **схему базы данных** – в ее структуру. Например, с их помощью можно добавлять новые таблицы, колонки, ограничения, процедуры и функции.

Сложно сформулировать требования к корректности использования **DDL** инструкций. В этом случае результатом (решением задачи) будет являться правильная схема БД, удовлетворяющая некоторым поставленным требованиям. Она может отличаться у преподавателя и студента, однако это не всегда означает, что студент построил схему неверно. Чтобы оценить правильность решения, преподавателю придется практически явно указать студенту в задаче, какие таблицы он должен создать и как их связать, иначе сложно оценить правильность выполнения автоматизированным способом. Однако такой подход снижает полезность заданий на проектирование БД, поскольку студент не проявляет собственного творчества.

DML [5] инструкции позволяют внести изменения в содержащиеся в БД данные – изменить значение в строке, отфильтровать набор данных по условию, сгруппировать и

отсортировать значения. Иными словами, они не изменяют структуру базы данных (схему базы данных).

Проверку решения задания по **DML** осуществить несколько проще, поскольку требования для DML проще сформулировать. Например, они могут быть сформулированы следующим образом.

- «*Необходимо написать запрос, который возвращает таблицу с колонками X, Y, Z… Таблица должна содержать данные, которые …*» Проверка для такой задачи заключается в построчной сверке таблицы, полученной в результате выполнения запроса студента, с таблицей, полученной в результате выполнения запроса преподавателя.
- «*Необходимо изменить данные в таблице, которые соответствуют определенным условиям (Инструкции INSERT, DELETE, UPDATE, MERGE и т.д.)*». Такую задачу можно проверить либо сверкой таблиц студента и преподавателя (в таком случае нужно получить таблицу преподавателя, выполнив его скрипт на копии той же БД), либо преобразовать этот тип задачи в первый тип задачи DML (в этом случае можно сверить результаты запросов SELECT студента и преподавателя, выполненных на измененных данных).

Предметом настоящего исследования является последняя из указанных подзадач, поскольку ее требования более прозрачны и понятны. Основной фокус в работе делается **на инструкциях DML**.

Одной из важных сторон рассматриваемой задачи является необходимость **ограничения возможностей студента** при решении задачи на написание запроса**.**

Требуется установить некоторые границы для запросов, чтобы при решении задачи пользователи или недоброжелатели не написали в запросе что-то, не предусмотренное автором. Например, необходимо, чтобы был закрыт доступ к системным базам, иначе можно повредить всю систему. Также пользователям должен быть закрыт доступ к другим БД на СУБД, не имеющим отношения к его задаче.

Данную проблему можно решить, используя профили безопасности. Автор задания при создании тестовой БД вводит код для создания пользователя базы данных, через которого будет работать студент, и список его прав. Пример выдачи разрешения пользователю «Larry» на запросы, содержащие SELECT и INSERT, для таблицы Employee:

```
GRANT SELECT, INSERT ON HumanResources.Employee TO Larry;
```

Таким образом, профиль ограничивает возможности пользователя при написании и выполнении запросов, тем самым повышая безопасность СУБД.

На данный момент нам известны несколько **аналогичных** систем для обучения, которые предоставляют возможность решать задачи с автоматической проверкой результата.

Одной из таких систем является **sql-ex.ru [6].** Эта система с закрытым исходным кодом позволяет самостоятельно обучаться языку SQL и решать типовые задания на написание запросов. Пользователю дается возможность прочесть задание и посмотреть схему базы данных, затем ввести свое решение в специальное текстовое поле и отправить его на проверку. Задание привязано к типу СУБД.

Другим примером может служить **olymp.krsu.edu.kg [7]**. Эта система тестирует правильность выполнения заданий для императивных языков программирования [8] методом черного ящика [9]; на вход подается поочередно серия тестовых данных и проверяется выход программы.

II. Алгоритм проверки решения

Нами предложен следующий алгоритм проверки решения.

1) Студент выбирает предметную область.
2) Студент выбирает задачу из предметной области.
3) Студент изучает материал, предоставленный автором задачи (ссылки на учебный материал, описание задачи, диаграмму БД и т.д.)
4) Студент пишет скрипт-решение в специальное поле, выбирает тип СУБД, на которой нужно выполнить скрипт, и нажимает «отправить».
5) Система получает решение, тип СУБД, и код задачи от студента.
6) Система отправляет на соответствующие типу СУБД подключенные экземпляры СУБД скрипт студента и скрипт автора задачи. Экземпляры возвращают результаты запросов системе.
7) Система в соответствии с правилами, поставленными преподавателем (например, проверить порядок строк), сверяет результат студента и автора.
    a) Если результаты совпадают (с учетом правил), то решение считается верным.
    b) Если по некоторым причинам, скрипт не удалось выполнить на БД, то студент получает сообщение, которое возвратила СУБД (ошибку).
    c) Если результаты студента и автора не совпадают, то студент получает результат своего запроса (если это разрешено автором) и результат автора (если это разрешено автором). На основании результатов он может изменить свой скрипт-решение.

Правила позволяют проверять разные детали решения, такие как сортировка данных, порядок столбцов, имена столбцов, отсутствие запрещенных ключевых слов.

Более подробно процесс проверки решения на сервере отображен на Рисунок 1.

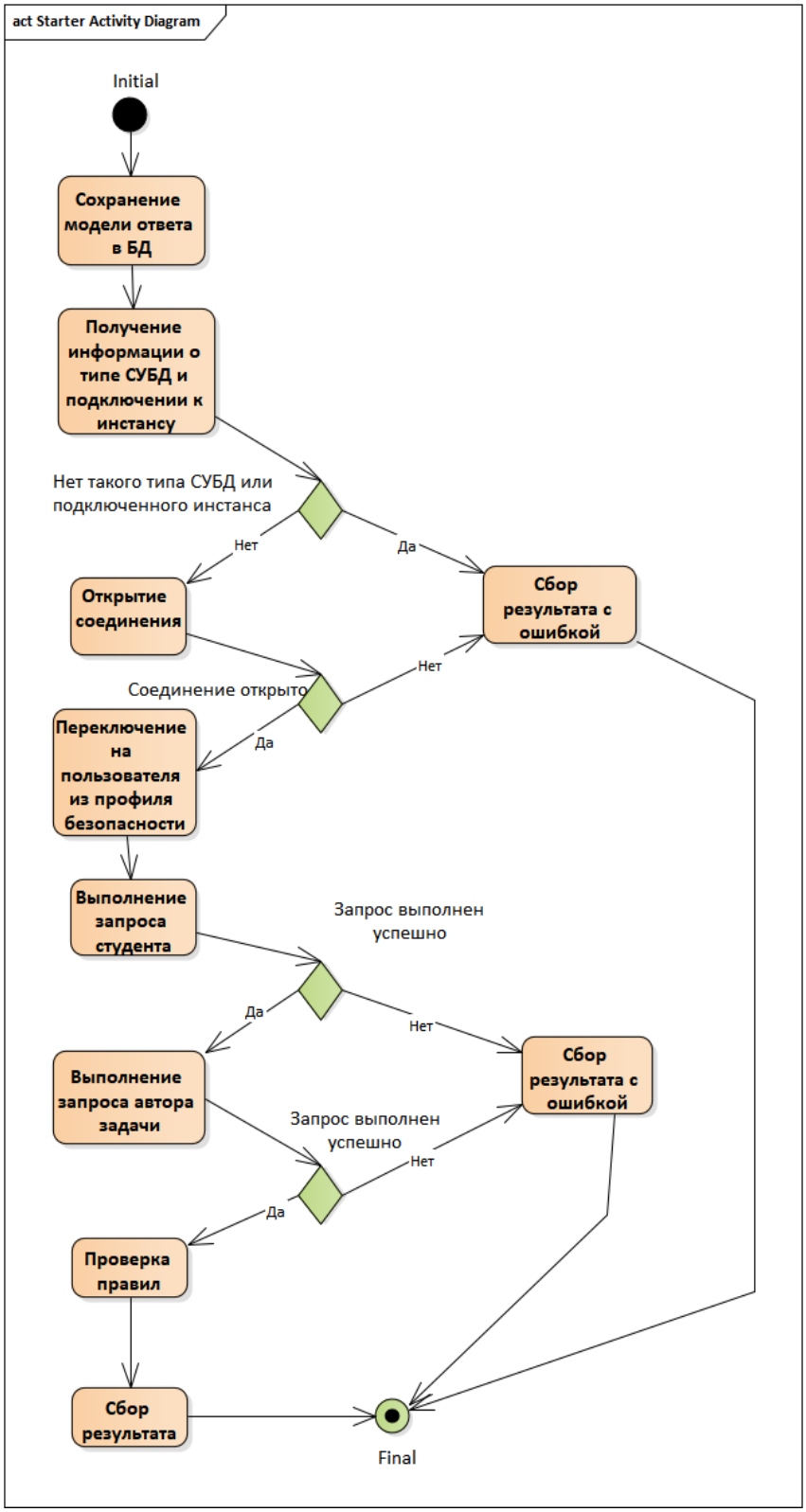


Рисунок 1 – Проверка решения на СУБД

Стоит обратить внимание на то, что первым выполняется скрипт студента, поскольку, очень вероятно, что скрипт будет с ошибками и может не выполниться. В этом случае не имеет смысла выполнять скрипт автора задания.

На основе предложенного алгоритма составлена альфа версия системы на ASP.NET MVC [88], которая имеет следующие возможности[1]:

- Автор задач может создавать абстрактную БД и заполнять ее данными;.
- Автор задач может создавать задачи в системе, внеся их описание, картинки диаграмм БД и свое эталонное решение к ним;
- Студент может изучить выбранную задачу из списка и решить ее, внеся в специальное поле SQL скрипт-решение и выбрав тип СУБД (пока только MS SQL);
- Система проверяет решение студента и выдает результат с подчеркиванием строк в результирующей таблице (ах), в которых допущена ошибка, выявленная при сравнении с ответом автора.

На Рисунок 2 показаны основные классы для выполнения команд на БД и проверки результата.

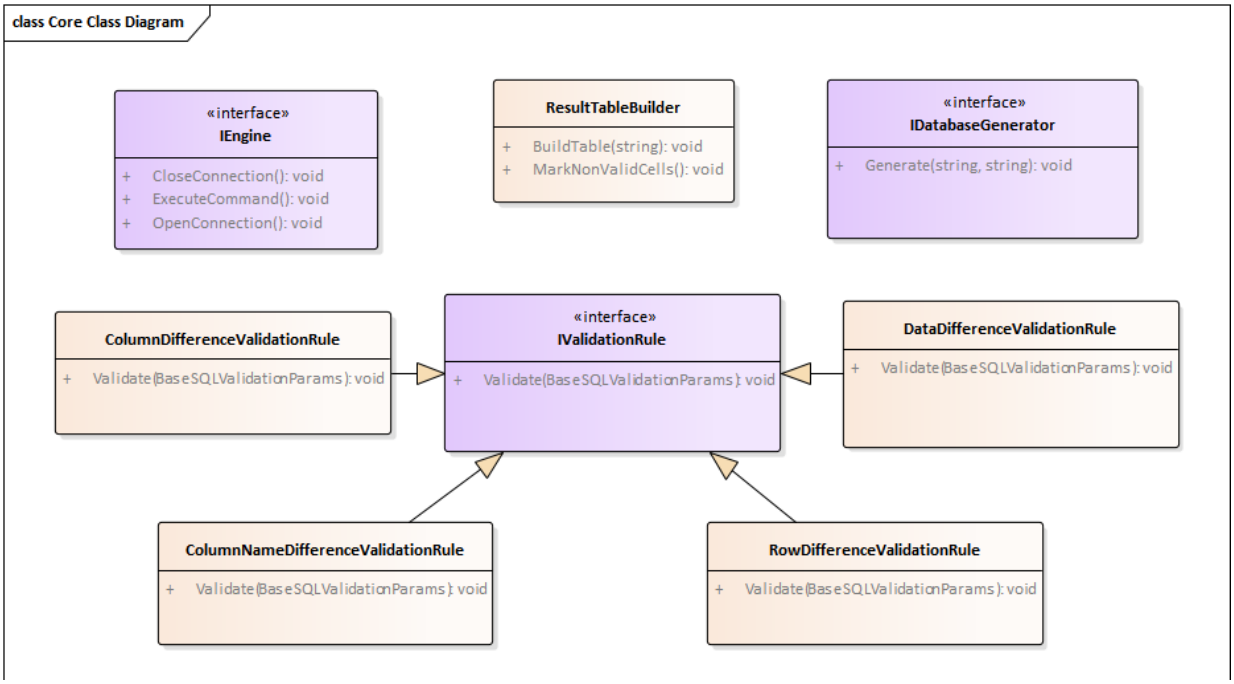


Рисунок 2 – Основные классы и интерфейсы для проверки решения

[1] Данная система была реализована как внутренний проект компании ОсОО «Таймлисофт» (Timelysoft LLC). Сайт компании: http://www.timelysoft.net/

ResultTableBuilder – строит таблицы с результатами (в виде моделей). Неправильные строки маркируются (отмечается поле IsValid = false в объекте строки).

IEngine – интерфейс для выполнения команд на СУБД. Классы, реализующие этот интерфейс, зависят от СУБД, с которой они работают и реализуются в отдельной библиотеке.

IDatabaseGenerator – интерфейс для выполнения команд по созданию и заполнению тестовыми данными БД. Классы, реализующие этот интерфейс, зависят от СУБД, с которой они работают и реализуются в отдельной библиотеке.

IValidationRule – интерфейс для правил проверки решения. Все классы, реализующие данный интерфейс, получают таблицы от запросов студента и автора для своей проверки.

RowDifferenceValidationRule – сверяет количество строк в таблице.

ColumnNameDifferenceValidationRule – сверяет имена колонок в таблицах автора и таблицах студента.

ColumnDifferenceValidationRule – сверяет количество колонок в таблицах автора и таблицах студента.

DataDifferenceValidationRule – сверяет данные в таблицах автора и таблицах студента, также может учитывать порядок строк.

Заключение

Предложен подход по автоматизации проверки результата запросов SQL, базирующийся на использовании DML (Data Manipulation Language) инструкций, с возможностью проверки решения одного и того же задания на разных реализациях СУБД.

Реализована альфа версия системы автоматизации проверки результатов запросов SQL, позволяющая автоматически проверять решение студента без необходимости вмешательства со стороны преподавателя, тем самым сокращая время, затрачиваемое на проверку задания.

Помимо MS SQL, система может также получить поддержку работы с другими СУБД, такими как PostgreSQL, MySQL. Поскольку результатами выполнения запросов в целом являются обыкновенные таблицы с данными, это позволяет студенту выбирать, на каком типе СУБД он хочет выполнить скрипт, не привязываясь к типу СУБД, на котором решил задачу автор. Достаточно будет сверить получившиеся таблицы от двух разных СУБД, чтобы определить – решена ли задача.

Литература

1. Что такое язык SQL (Structured Query Language). URL: https://ru.wikipedia.org/?oldid=103455867 (дата обращения 1 декабря 2019).
2. Определение «электронное обучение». URL: https://ru.wikipedia.org/?oldid=103020624 (дата обращения 29 октября 2019).
3. Определение «Система управления базами данных». URL: https://ru.wikipedia.org/wiki/Система_управления_базами_данных.
4. Data Definition Language (на англ.). URL: https://en.wikipedia.org/w/index.php?title=Data_definition_language&oldid=923093270 (дата обращения 1 декабря 2019).
5. Data Manipulation Language (на англ.). URL: https://en.wikipedia.org/w/index.php?title=Data_manipulation_language&oldid=915843955 (дата обращения 1 декабря 2019).
6. Сайт для самостоятельного изучения SQL. URL: http://sql-ex.ru (дата обращения 1 декабря 2019).
7. Сайт для проведения олимпиадных соревнований и тренировки по программированию университета КРСУ. URL: http://olymp.krsu.edu.kg/ (дата обращения 1 декабря 2019).
8. Императивное программирование и императивный язык программирования. URL: https://ru.wikipedia.org/?oldid=101934922 (дата обращения 1 декабря 2019).
9. Тестирование по стратегии черного ящика. URL: https://ru.wikipedia.org/?oldid=86881359 (дата обращения 1 декабря 2019).
10. Альфа версия системы SolveWay. URL: https://www.solveway.club

УДК 004.415.2 10.5281/zenodo.3904148

Д. Авельцов, В. Гайдамако, *dolpha@gmail.com*
Институт машиноведения и автоматики НАН КР, Бишкек

РАЗРАБОТКА МОДУЛЯ ВИРТУАЛИЗАЦИИ СЕНСОРНЫХ УСТРОЙСТВ ДЛЯ РАСПРЕДЕЛЕННЫХ ИНФОРМАЦИОННО-ИЗМЕРИТЕЛЬНЫХ СИСТЕМ

В настоящей работе рассмотрен пример разработки модуля виртуализации сенсорных устройств для использования в Интернете вещей, распределенных и облачных информационно-измерительных системах, предоставляющих измерительные устройства в качестве услуги. При реализации системы использовались стандарты описания датчиков OGC. Также на основе данных стандартов имплементированы модели данных и связи между ними. Разработан модуль взаимодействия с пользователем и проведено тестирование.

Ключевые слова: сенсорные устройства, датчики, виртуализация, микросервисы. виртуализация датчиков, docker, docker-контейнер, микросервисы (microservices), OGC (Open Geospatial Consortium), sensorML, Интернет вещей, информационно-измерительные системы (ИИС)

Введение

В современном мире, мире Интернета Вещей (ИВ), умных домов и умных городов, умные устройства – сенсоры и актуаторы, используются широко и повсеместно, и стремительный рост их числа приводит к росту потребности в решениях, позволяющих подключать эти устройства к сети, собирать, хранить и анализировать их данные. Применение облачных технологий позволяет различным пользователям и приложениям получать совместный (разделяемый) доступ к устройствам и к их данным через технологию виртуализации [1-6]. Для физического устройства создается множество его виртуальных представлений, свое для каждого конкретного пользователя. Виртуальное устройство может отображать не одно, а комбинации нескольких физических устройств в зависимости от запросов пользователей. Использование виртуальных устройств вносит в систему дополнительный слой абстракции, что позволяет масштабировать систему и работать с устройствами, не заботясь об особенностях их физической реализации. Виртуализация датчиков используется в программно-аппаратных системах, реализующих подходы Sensor-Cloud Infrastructure (SCI), Information-as-a-Service (IaaS), Framework of Sensor-Cloud Integration (FSCI), Virtual Federated Sensor Network (VFSN), Sense-Cloud [5].

При организации совместного доступа важна стандартизация – стандарты описания и управления устройствами, стандарты для форматов и структуры данных, генерируемых этими устройствами, становятся острой необходимостью. Было создано множество стандартов для описания устройств. Одной из организаций, занимающихся стандартизацией, является OGC – Open Geospatial Consortium (Открытый Консорциум Геопрстранственной информации), международная некоммерческая организация, созданная в 1994 году для разработки стандартов геопространственных данных, учета и согласования координатных систем, используемых в мире. В настоящее время координирует деятельность более 500 [7] правительственных, коммерческих, некоммерческих и научно-исследовательских организаций с целью разработки и внедрения консенсусных решений в области открытых стандартов не только для геопространственных данных, обработки данных геоинформационных систем, но и

других стандартов в различных областях, особенно там, где важны данные о положении в пространстве. Стандартами OGC, связанными с описанием и использованием сенсорных устройств, являются Semantic Sensor Network Ontology (Стандарт описания сенсорных сетей), SensorML 2.1 – язык моделирования сенсорных устройств. SensorML предоставляет модели и XML-кодировки для описания сенсоров, актуаторов и процессов, позволяет реализовать описание спецификации обслуживания устройства, описание входных данных, выполнение агрегатных процессов и рабочих процессов по требованию и потоковую передачу данных. Стандарты ISO [8] используют OGC стандарт gml (Geography Markup Language) для описания устройств и данных.

В данной статье рассматриваются вопросы разработки программных средств виртуализации сенсорных устройств для создания виртуальной сенсорной среды и изоляции приложений-клиентов от оборудования.

Инфраструктура распределенной/облачной информационно-измерительной системы

Стандартная модель инфраструктуры распределенной/облачной Информационно-Измерительной Системы (ИИС), которая используется и в Интернете Вещей (ИВ, англ. Internet of Things, IoT), представлена на рисунке 1 [1, 5]. Данные передаются от сенсорных устройств, датчиков различных типов, объединенных в Беспроводные Сенсорные Сети (БСС) к концентраторам (в литературе также может иметь названия: шлюз, хаб – устройство для связи с глобальной сетью, может исполнять различные дополнительные функции, в том числе выступать в качестве частного облака), далее информация передается на облачную платформу, которая также может быть интегрирована с другими сервисами/платформами. Соответствующие модули обеспечивают связь с пользовательскими приложениями на различных типах устройств – стационарных десктопах, мобильных устройствах, специализированных компьютерах.

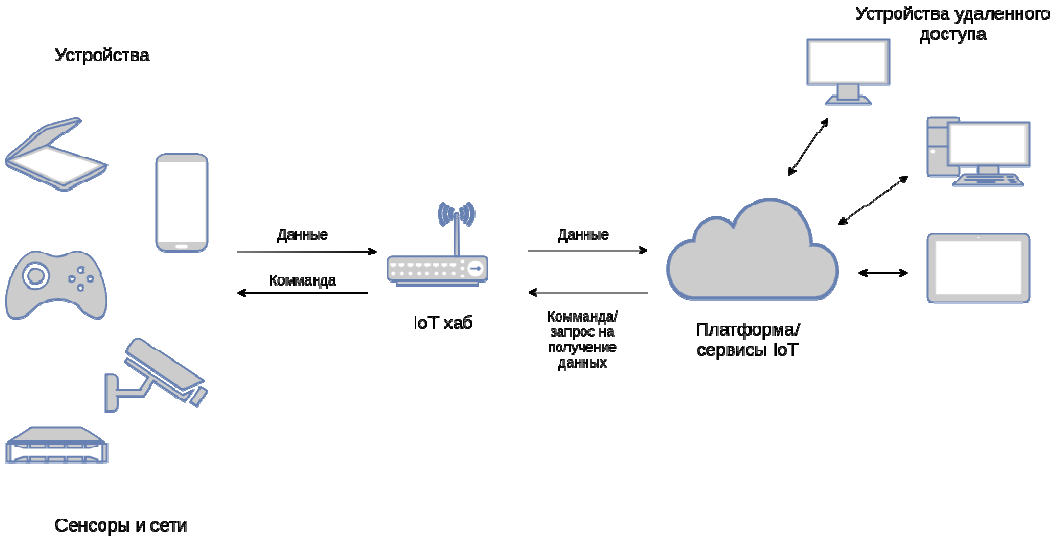


Рисунок 1 – Инфраструктура Интернета вещей

Упрощенная модель архитектуры сети распределенной/облачной ИИС представлена на рис. 2. Типы сетей для каждого уровня зависят от реализации устройств – участников данной сети, а также от иных факторов, влияющих на возможность использовать те или иные типы сетей.

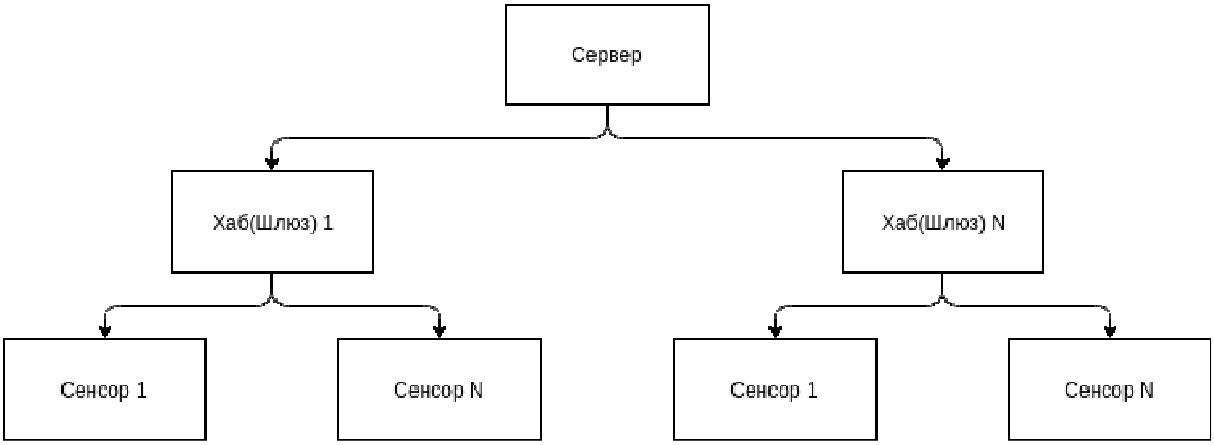


Рисунок 2 – Схема архитектуры сети распределенной ИИС

Общая архитектура решения для ИВ или (если исключить актуаторы) для распределенной/облачной ИИС представлена на рис. 3. Физические сенсорные устройства (датчики) регистрируют параметры окружающей среды, формируют аналоговый сигнал, который АЦП (аналого-цифровой преобразователь) преобразует в цифровой. После конвертации информация обрабатывается локальным процессором периферийного устройства, в частности, ставится метка-идентификатор (тег), обязательно включающий временную метку(timestamp).

Следующее звено – коммутатор (хаб) может находиться как в облаке, так и на периферии. Коммутатор перенаправляет полученную информацию на различные объекты, используя теги. В роли объектов могут выступать различные сервисы, очереди или хранилище. Так происходит получение и обработка информации от конкретного периферийного устройства. Далее на уровне интегратора полученная информация от различных периферийных устройств суммируется по однотипным тегам, при этом типы устройств могут быть различными. Информация от всех объектов систематизируется аналитическим блоком, который может включать Искусственный Интеллект (ИИ), машинное обучение и прочее. Блок презентации отвечает за взаимодействие с пользователем системы.

Любая распределенная система требует механизмов гарантированной доставки информации. В блок виртуального представления периферийного устройства – ту часть, которую представляет собой разрабатываемая система, записывается информация, принятая (или передаваемая в случае актуатора) от периферийного устройства. В случае неудачной попытки передачи информации от периферийного устройства в облако, либо от облака к периферийному устройству, осуществляются повторные попытки передачи и кэширование данных на уровне виртуального устройства.

Требования к системе

Целью представляемой работы было создание прототипа модуля виртуализации датчиков для распределенной/облачной информационно-измерительной системы и ИВ и безотказной передачи данных в сетях различной топологии.

Разрабатываемая система может применяться для виртуализации датчиков, для получения информации, собранной датчиками, для получения информации о датчиках, а также поиска информации и датчиков.

Модуль виртуализации датчиков должен быть независимой системой, предоставляющей пользовательский интерфейс для описания и управления датчиками, а также API (Application Programming Interface, программный интерфейс приложения) для любых авторизованных клиентских приложений.

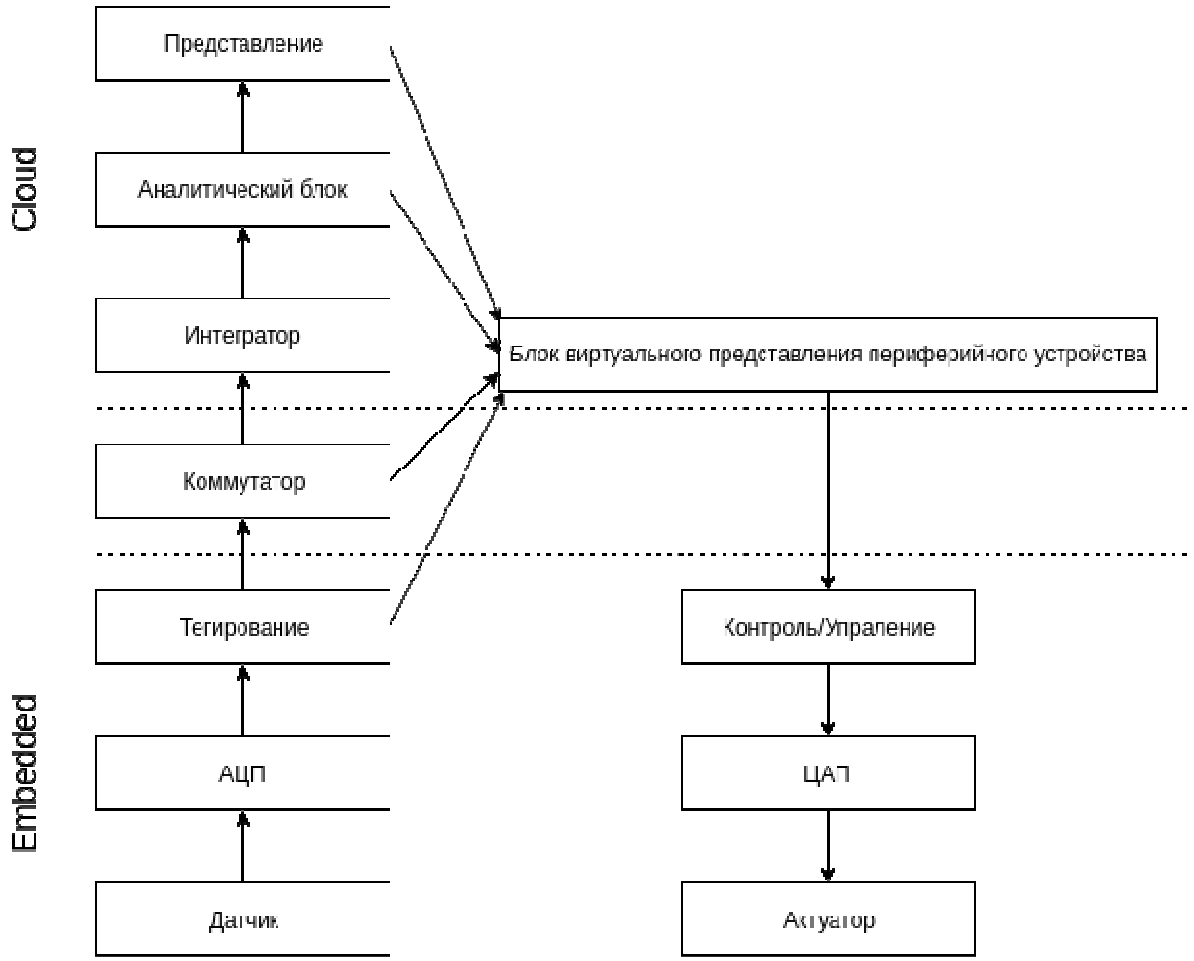


Рисунок 3 – Общая архитектура решения для распределенной системы. Embedded – реализация на стороне устройства, Cloud – реализация в облачной инфраструктуре

Система должна выполнять следующие функции:

- Регистрация / авторизация пользователя;
- Создание, редактирование, экспорт шаблонов на языке SensorML при помощи веб-интерфейса;
- Виртуализация, создание виртуального представления датчиков на основе шаблона SensorML;
- Обработка запросов от датчиков;
- Получение данных с физических датчиков через слой виртуализации;
- Хранение информации, полученной от датчиков; хранение информации должно происходить как на уровне периферийного устройства, так и в облаке. Периферийное устройство сохраняет свои настройки, состояние и временно хранит информацию (кэширует), пока данные гарантированно не переданы в облако
- Кэширование информации, полученной от датчиков;

Система разрабатывается для 3 групп пользователей:

- Администратор системы. Имеет общее представление об аспектах использования системы;
- Владелец датчика. Должен знать технические характеристики своего датчика, быть способным описать его;
- Пользователь (клиент) системы. Особые требования не предъявляются;

Система должна предоставлять следующие интерфейсы:

- Пользовательский интерфейс для создания шаблонов датчиков и управления датчиками для владельца устройства;
- Программный интерфейс для получения информации о датчиках и информации, собранной датчиками;

Функциональные требования к системе отражены на диаграмме, представленной на рисунке 4 (Диаграмма вариантов использования).

Внешние требования к интерфейсам

Взаимодействие с пользовательской частью системы осуществляется с помощью HTTP-запросов.

Взаимодействие датчиков с системой осуществляется при помощи протоколов HTTP/S, TCP, HTTP 2.

На серверной машине установлена операционная система семейства linux, компилятор языка go версии 1.14 или выше, Docker Engine версии 19.03 или выше; на клиентской и серверных машинах есть доступ к сети Internet;

Безопасность. Схема распределения ключей шифрования – главная особенность, отличающая сети ИВ от обычной mesh-сети. Все устройства являются крайне недоверенными – любое устройство может быть скомпрометировано, поэтому ни одно устройство не может хранить мастер-ключ всей системы.

- Включение нового устройства в сеть должно происходить децентрализовано (на случай отсутствия связи с облаком).
- Необходимо использование протоколов, использующих схемы разделения секрета и слепой подписи.
- Система должна хранить логи и историю работы всех компонентов программной системы за последний месяц;
- Любые сбои в программной системе должны протоколироваться;
- Система должна быть защищена от несанкционированного доступа к данным и коду;
- Каждый датчик должен иметь свой уникальный идентификатор, который позволит идентифицировать его в системе;
- Каждый датчик имеет свою уникальную конфигурацию, которая описана в шаблоне, общее представление о подлинности датчика должно складываться из его идентификатора и данных его конфигурации;

Стандартизация. Система должна обслуживать различные устройства, компоненты и процессы. Для стандартизации описания требуется выбрать утвержденный стандарт описания устройств. SensorML (Sensor Model Language) является стандартом, одобренным Открытым консорциумом геопространственных данных – Open Geospacial Consortium (OGC) [7]. SensorML обеспечивает стандартные модели и в XML-кодирование [9] для описания датчиков и измерительных процессов. SensorML может быть использован для описания широкого спектра датчиков, в том числе динамических и стационарных платформ, и как in-situ, так и дистанционных датчиков.

Подходы к проектированию системы

В разрабатываемой системе используется как монолитный, так и микросервисный подход в проектировании. Пользовательская часть написана с применением монолитного подхода. Основное ядро системы спроектировано с применением

микросервисного подхода [10]. Такое решение продиктовано необходимостью масштабируемости системы. также через микросервисы можно объединять разные технологии, выбирая лучшие из возможных решений. Так как микросервисы являются независимыми друг от друга, стабильность системы повышается. Сбои и дефекты в одном микросервисе не повлияют на работу остальных, поэтому сама система будет функционировать с минимальными простоями. Всю систему условно можно поделить на модули (рис. 5): веб-сервер (Web-Server) и пользовательский интерфейс, платформа виртуализации (микросервисы) и виртуальные сенсорные узлы (Virtual Sensor Nodes).

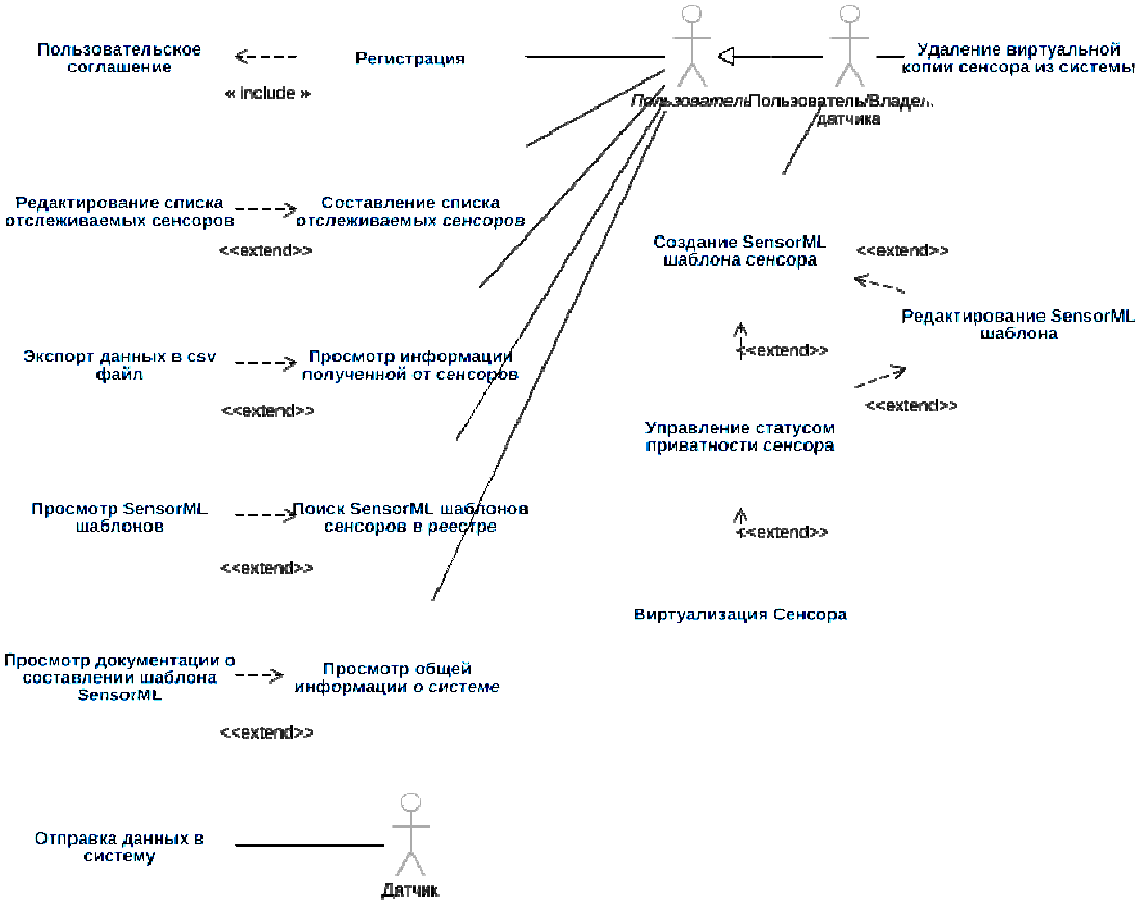


Рисунок 4 – Функции системы виртуализации сенсорных устройств
(Варианты использования)

Пользовательский интерфейс и веб сервер (Web Server)

Пользовательский интерфейс написан на языке typescript с использованием фреймворка Angular 7. Компоненты пользовательской части:

- api. Отвечает за взаимодействие с серверной частью. Взаимодействие происходят в соответствии со стандартом REST;
- editor. модуль SensorML редактора;
- model. модели стандарта SensorML;
- create. модуль загрузки и создания SensorML шаблона;
- services. Общие сервисы приложения;
- user. Пользовательский модуль, отвечает за регистрацию и авторизацию;

Для взаимодействия с серверной частью было решено использовать REST (REpresentational State Transfer) [11] архитектуру. Архитектура REST разработана для соответствия протоколу HTTP [12], используемому в сети Интернет. Вызов удаленной процедуры представляет собой обычный HTTP-запрос («REST-запрос»), а необходимые данные передаются в качестве параметров запроса.

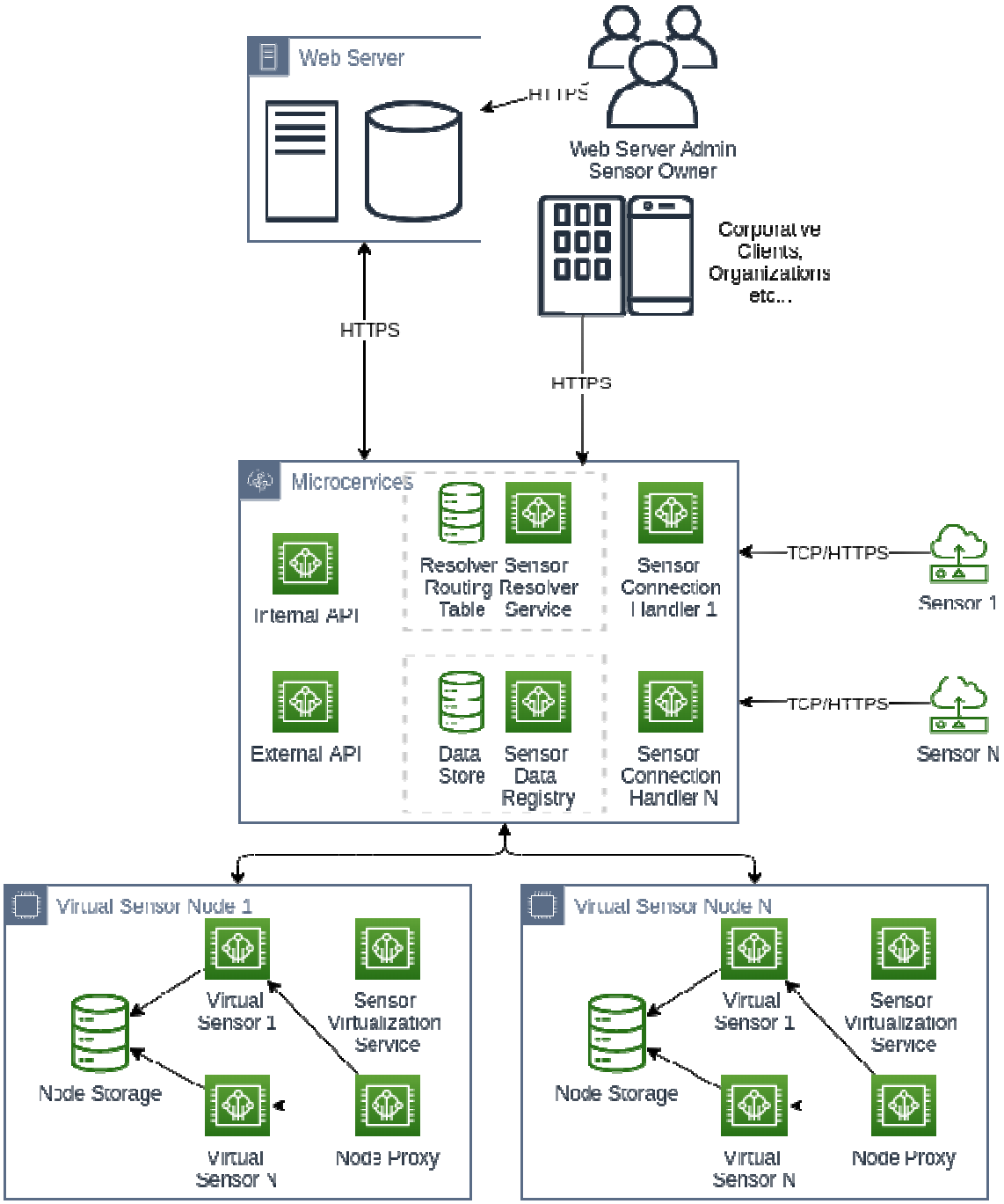


Рисунок 5 – Общая архитектура системы виртуализации сенсорных устройств

Архитектура REST очень проста в плане использования. Данные, передаваемые в теле запроса, могут быть в XML [9], JSON [13] формате, или же они могут быть переданы с помощью аргументов в URL (Universal Resourse Locator). В

разрабатываемой системе решено использовать JSON формат для изменения состояния системы и аргументы URL для методов получения данных.

Серверная часть написана на языке PHP 7.4 с использованием фреймворка Symfony 4.4. Данный компонент системы обслуживает пользовательский интерфейс и перенаправляет запросы виртуализации в платформу.

Архитектура Web Server содержит три слоя:

- слой доступа к базе данных, основанный на объектно-ориентированном отображении (ORM, Object Relational Mapping), содержит классы-сущности, отображающие состояние базы данных в состояние объектов этих классов-сущностей;
- слой бизнес-логики, содержащий всю логику обработки и изменения сущностей, проецируемых в базу данных;
- слой контроллеров, представляющий собой контроллеры для обработки запросов; представления определяются форматом данных, используемым для обращения к контроллерам.

Основные классы-сущности – это классы User и Sensor. Класс User представляет информацию о зарегистрированном пользователе, класс Sensor – информацию о созданном датчике, его шаблон, статус приватности, статус виртуализации.

Слой бизнес-логики описывает алгоритмы создания, получения, изменения, удаления для каждой сущности, а также алгоритмы обработки данных, представленных объектом сущности. Эти алгоритмы содержатся в классе, соответствующем классу-сущности, который они обрабатывают. В качестве параметров и возвращаемых результатов классы бизнес-логики используют экземпляры классов-сущностей.

Слой контроллеров представляет собой граничные классы, предоставляющие API интерфейс. Каждый граничный класс содержит методы обработки запроса. Запросы соответствуют сценариям использования системы. Основные граничные классы системы представлены на рис. 6. Методы этих классов должны извлекать параметры запроса, использовать методы бизнес-логики для обработки данных и возвращать результаты обработки в соответствующем JSON-формате. Никакой логики обработки данных контроллеры содержать не могут.

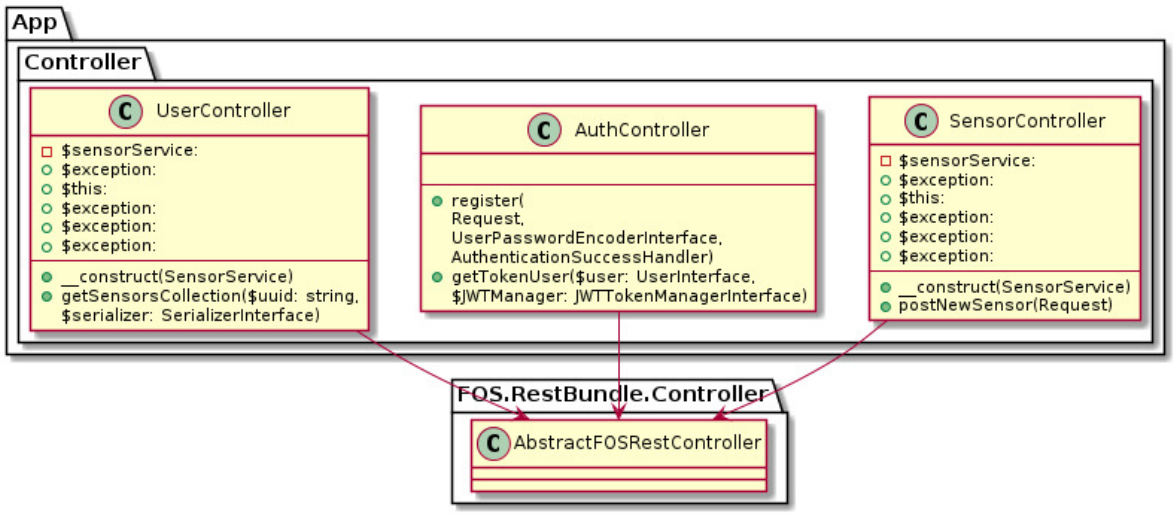


Рисунок 6 – Слой бизнес логики. Контроллеры (Диаграмма классов)

В качестве параметров и возвращаемых результатов граничные классы используют данные в формате JSON или полученные аргументы URL-запроса.

Диаграмма развертывания пользовательского модуля интерфейса и веб сервера (Web Server) описывает физические узлы в наиболее типичной конфигурации для поставщика услуг “виртуализация датчиков для распределенных измерительных систем” (рис. 6):

- Load Balancer Server – Балансировщик нагрузки распределяет трафик между серверными узлами веб-приложения, когда запущено более одного экземпляра сервера. К балансировщику нагрузки нет никаких особых требований. В качестве примера используется обратный прокси-сервер nginx.
- Web Application Server – Сервер веб приложения, обеспечивает пользовательский доступ к общей информации о системе, созданию шаблонов сенсоров при помощи графического интерфейса, поиску шаблонов сенсоров, просмотру информации о сенсорах, получаемой измерительной информации от сенсоров. Также обеспечивает администратору системы доступ через графический интерфейс для редактирования общей информации о системе и для администрирования пользователей, шаблонов и виртуальных сенсоров.
- Web application RDBMS Server – Сервер с установленным сервером баз данных. Используется для хранения данных пользователей и общей информации о системе для отображения в веб приложении.
- Sensor Template Registry Module / Sensor Virtualization Module – модуль реестра шаблонов датчиков / модуль виртуализации, не вошли в диаграмму на рис.7.

Пользовательский интерфейс используется для взаимодействия пользователя (владельца сенсора, администратора) с системой. Создание, редактирование и удаление шаблона сенсорного устройства на языке SensorML реализовано при помощи редактора шаблонов. В текущей версии системы пользователь может также объединить компоненты в виде датчиков в физические системы – например, описав термометр, анемометр и процесс вычисления фактора ветра (обычно при помощи него вычисляют температуру “как ощущается”), можно объединить все эти компоненты в погодную станцию и не собирать данные каждого сенсора в отдельности, а работать уже с физической системой которая объединяет в себе все эти компоненты.

Редактор SensorML написан на основе редактора smle-editor, автором которого является организация “52North”, большая часть исходного кода была переписана и обновлена до новой версии фреймворка Angular 7, поддержка редактора прекращена 2 года назад (2018) от текущей даты [14]. Бекенд часть позволяет управлять пользовательскими данными и данными о системе (документация, ознакомительная информация). Взаимодействие с платформой происходит через внутренний программный интерфейс платформы.

Проектирование и конструирование модуля платформы виртуализации (Microservices)

Данная платформа написана с применением подхода микросервисной архитектуры [10]. Использованы такие решения, как: golang, go-micro, gRPC, Protobuf, mongodb, docker. Платформа виртуализации содржит следующие микросервисы:

Sensor connection handler. Сервис обработки запросов от сенсорных устройств. Находится под нагрузкой, т.к. является основной точкой входа в систему. При необходимости для снижения общей нагрузки на систему можно поднять несколько копий на разных серверных машинах. Система с одним экземпляром данного сервиса способна обработать около 500 запросов в секунду от сенсорных устройств.

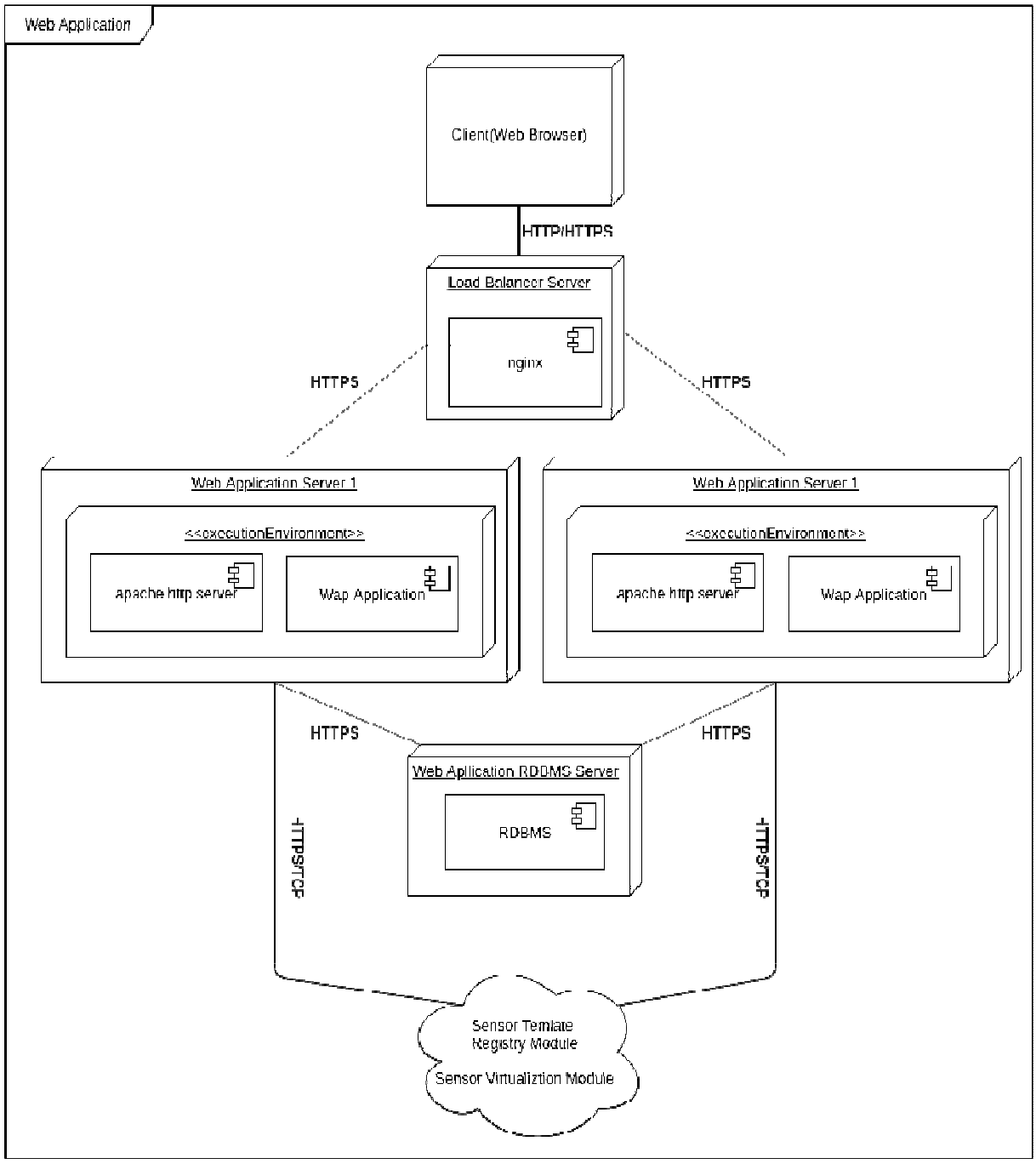


Рисунок 7 – Физические узлы веб приложения (диаграмма развертывания)

Sensor Resolver. Сервис маршрутизации сенсор – контейнер. По своему устройству напоминает работу свитча(switch) – устройства для маршрутизации в сети. Хранит соотношения сенсор – контейнер с учетом расположения узла – группы виртуальных датчиков и перенаправляет запросы от сервисов Sensor connection handler к узлам, где находится виртуальное представление датчика, пославшего запрос.

Sensor data registry. Общее хранилище (реестр) данных, полученных от всех устройств, зарегистрированных в системе. Является легковесным сервисом, хранение данных осуществляется при помощи базы данных mongodb.

Internal API. Программный интерфейс для взаимодействия бекенда веб-сайта с платформой.

External API. Программный интерфейс для взаимодействия с клиентами системы. Клиенты могут посылать запросы на получение данных за определенный период или подписываться на обновление данных.

Ogc-models. Библиотека с описанием моделей данных. Данная библиотека содержит в себе следующие стандарты OGC: GCO, GMD, GML, SAMS, SF, SML, SWE

Virtual sensor node. Группы виртуальных датчиков;

Node Proxy – сервис, который принимает запросы от Sensor Resolver и перенаправляет их на контейнеры, которые содержат в себе виртуальные представления устройств, данные контейнеры и прокси сервис находятся в одной виртуальной сети.

Sensor Virtual Instance – виртуальное представление устройства, создается на основе sensorML шаблона. Может представлять как физический компонент, так и физическую систему или процесс. Т.к. данный сервис имеет всю информацию о своей физической сущности, он “знает”, как интерпретировать информацию, полученную от физического устройства, кеширует ее и перенаправляет на сервис реестра информации полученной от устройств.

Виртуализация устройства из пользовательского интерфейса. Последовательность событий при виртуализации изображена на рис. 8. При нажатии на кнопку “виртуализировать”, запрос направляется на сервер, где берется шаблон выбранного датчика и запрос посылается в платформу, на внутренний программный интерфейс системы. Данный программный интерфейс перенаправляет запрос на сервис виртуализации. сервис виртуализации создает или пересоздает, если уже существует, контейнер с виртуальным представлением устройства на основе полученного шаблона и возвращает идентификационную строку по тому же маршруту пользователю.

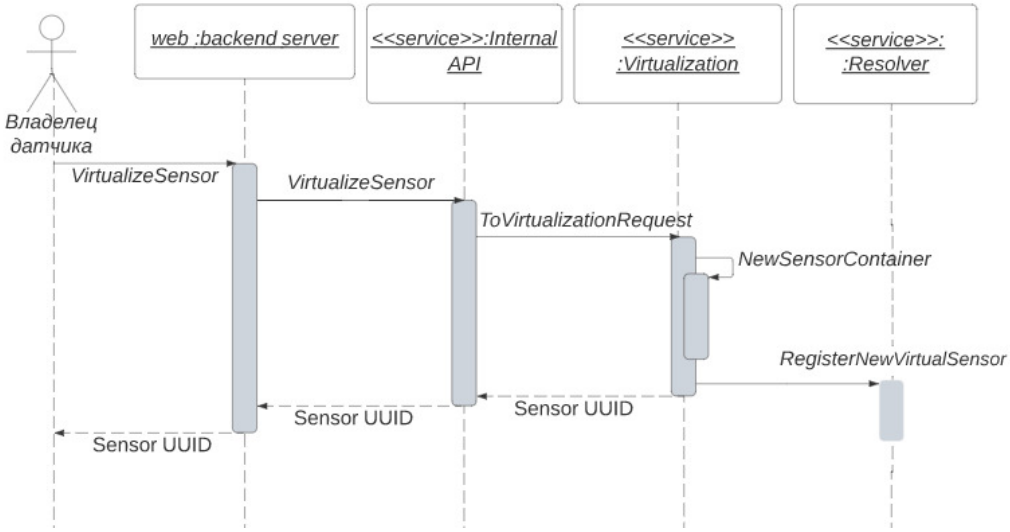


Рисунок 8 – Последовательность действий при запуске виртуализации датчика владельцем

Пользователь, используя эту строку, может идентифицировать свое устройство в системе. Для идентификации ему нужно следовать инструкции, которая будет показана ему вместе с идентификационной строкой. Далее сервис виртуализации добавляет созданный контейнер в виртуальную сеть узла, после чего сообщает резолверу о регистрации нового виртуального представления устройства и все его данные.

Диаграмма развертывания платформы виртуализации

Диаграмма развертывания архитектуры модулей платформы виртуализации датчиков (microservices) (рис. 9) и диаграмма развертывания архитектуры узла виртуальных сенсоров (рис. 10) описывают наиболее типичную конфигурацию

платформы для поставщика услуг “виртуализация датчиков для распределенных измерительных систем”.

Тестирование и оценка производительности прототипа системы виртуализации сенсорных устройств. При разработке системы проводилось тестирование компонентов (модульное тестирование), тестирование интеграции, системное тестирование и приемочные испытания. Приемочные испытания проводились для определения соответствия функциональным требованиям для веб-сервера и платформы виртуализации в соответствии со следующими сценариями:

- Создание сущности – тест проверяет создание сущности в системе.
- Частичное обновление сущности, тест проверяет обновление поля сущности.
- Получение информации, тест проверяет получение информации о физическом датчике и создание виртуальной копии датчика.
- Запрос к физическому устройству, тест проверяет результат запроса для передачи данных от физического датчика его виртуальной копии с последующим их сохранением.

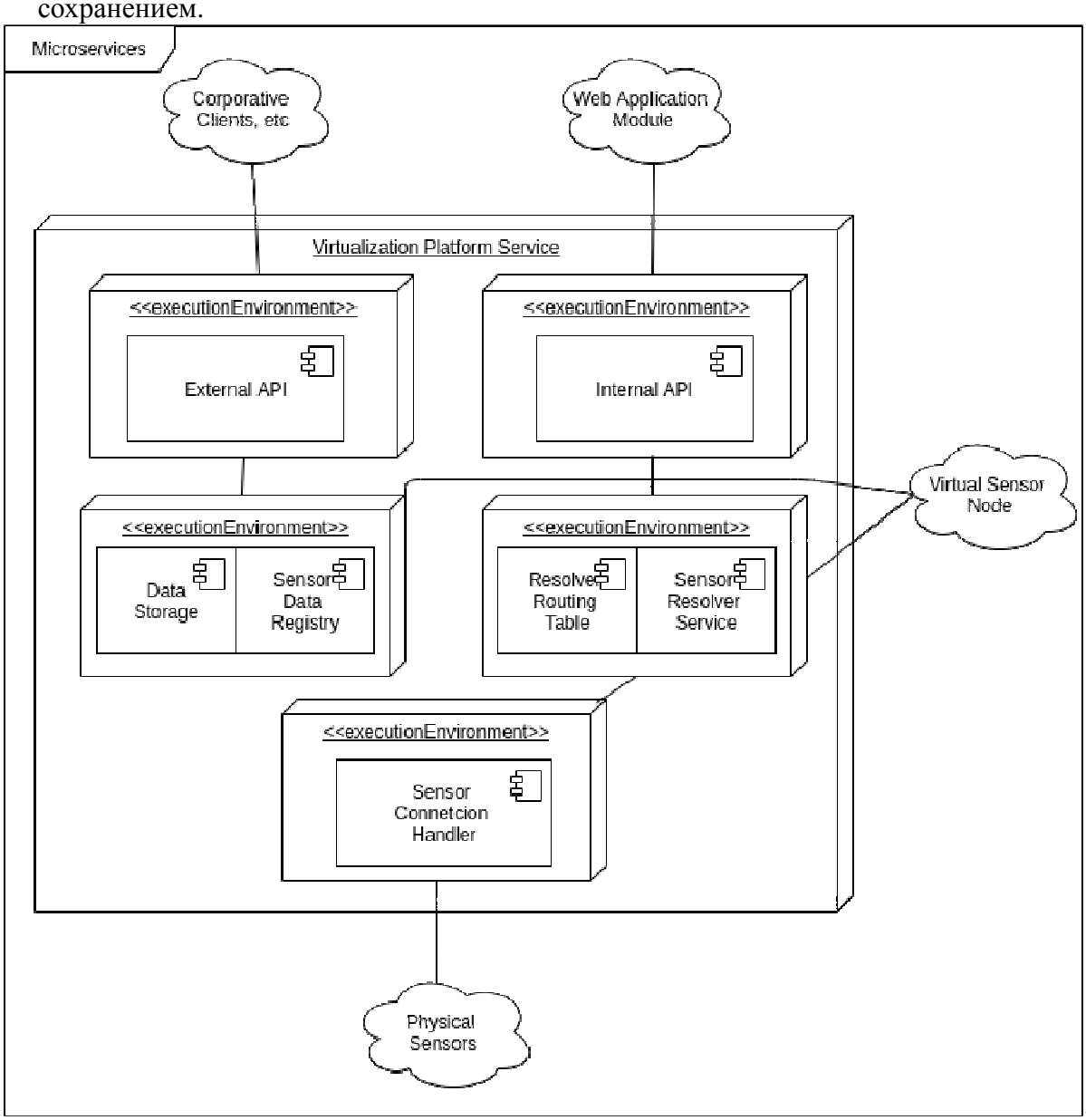


Рисунок 9 – Диаграмма развертывания архитектуры модулей платформы виртуализации датчиков.

При нагрузочном тестировании было выявлено, что один экземпляр сервиса sensor connection handler может выдержать не более 500 запросов в секунду. При возрастании нагрузки на данный сервис требуется установить еще одну копию сервиса и балансировать нагрузку между экземплярами сервиса.

Прототип системы тестировался на параметрах – процессор 2.0GHz, 8GB оперативной памяти и был запущен на десктопной системе с другими потребителями.

При нагрузке до 500 запросов в секунду система потребляет приемлемое количество ресурсов. Стоит сделать замечание, что качество канала связи между датчиком и системой не может быть известно заранее, что может повлечь за собой проблему медленно читающего клиента. Стоит рассмотреть внесение в систему возможности автоматического масштабирования точек входа для датчиков – sensor connection service.

Использование интерфейса не вызвало затруднений. Предъявленные к пользователю требования – знать характеристики своего датчика, являются оправданными. Редактор SensorML является интуитивно понятным для пользователя, знакомого с английским языком и устройством своего датчика.

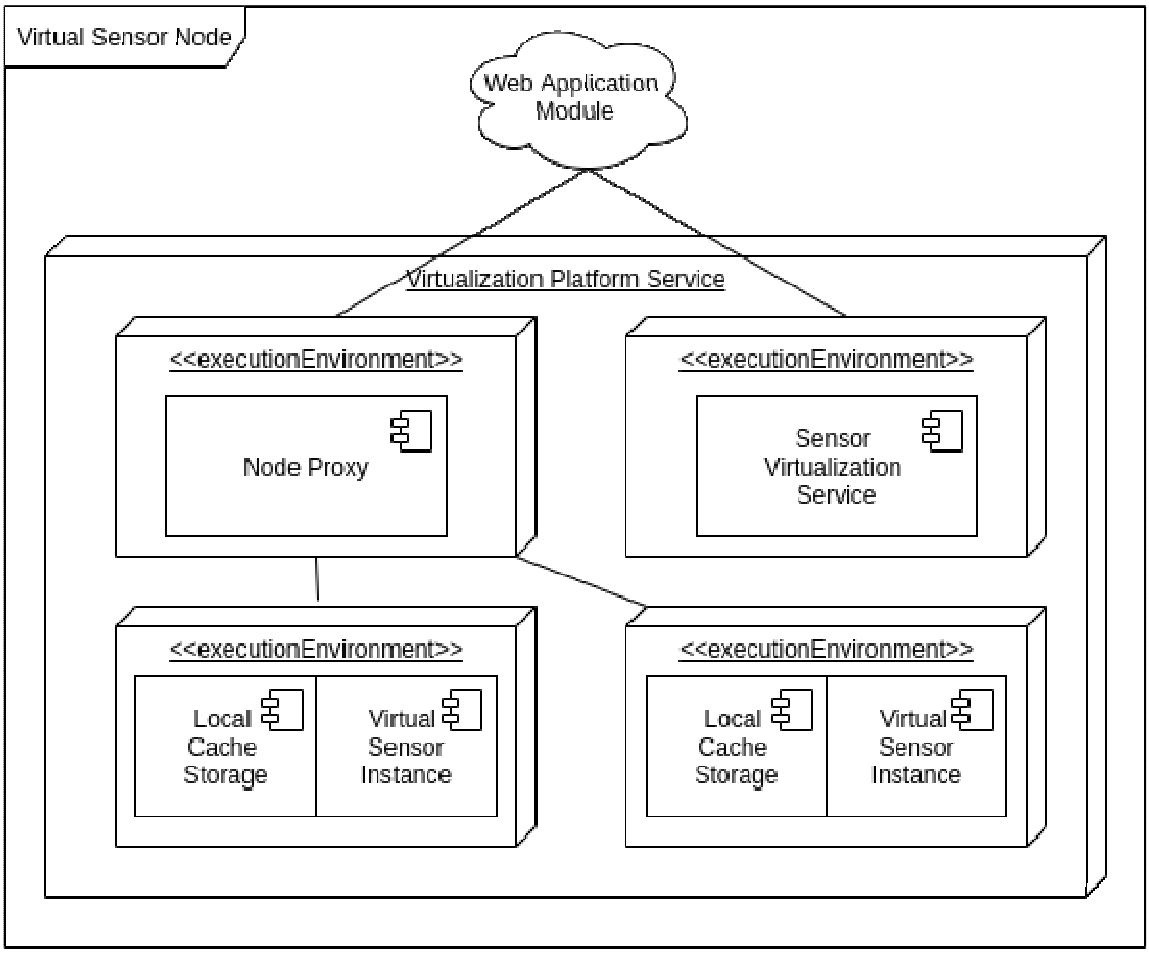


Рисунок 10 – Диаграмма развертывания архитектуры узла виртуальных сенсоров

Заключение

Был разработан и протестирован прототип системы виртуализации датчиков. Смоделирована архитектура системы. Для реализации системы виртуализации датчиков были выбраны языки: Golang, php7.4, typescript в сочетании с популярными

технологиями: go-micro, protocol buffers, grpc, docker. Предложенный подход по виртуализации сенсорных устройств позволил достичь следующего:

- Реализована виртуализация датчиков;
- Система реализует стандарт SensotML и позволяет пользователям легко описать свои устройства, используя веб форму.
- Система обслуживает большое количество запросов (до 500 запросов в секунду на одну точку входа при конфигурации сервера 4Гб оперативной памяти и двухъядерном процессоре с частотой 2.0 GHz).
- Система обеспечивает постоянный доступ к виртуальным представлениям устройств и их данным.
- Система может принимать данные любых типов устройств, описанных при помощи стандарта SensorML.
- Система скрывает физические устройства за слоем абстракции, таким образом, что пользователи не имеют неограниченного доступа к устройствам и не провоцируют разряд батареи устройства высоким количеством запросов, но могут полноценно взаимодействовать с их виртуальными копиями.
- Система позволяет пользователю не обращаться на адрес в сети каждого из устройств, а иметь их представления на одном узле, что облегчает работу с адресацией устройств в сети для пользователя

В дальнейшем планируется наращивать функционал системы для виртуализации датчиков и улучшать возможности масштабирования системы.

Система была использована для исследования эффективности виртуализации датчиков. С ее помощью были проведены эксперименты с использованием технологий виртуализации контейнеров и сетей. Проведен анализ эффективности работы виртуальных устройств с использованием контейнеров, подключаемых к виртуальным сетям и передачи данных между глобальной сетью и слоями виртуальных сетей, получены результаты и описаны дальнейшие перспективы исследования.

Литература

1. Madoka Yuriyama, Takayuki Kushida, Sensor-Cloud Infrastructure – Physical Sensor Management with Virtualized Sensors on Cloud Compputing. 13th International Conference on Network-Based Information Systems 2010.
2. Khan, I.; Errounda, F.Z.; Yangui, S.; Glitho, R.; Crespi, N., – Getting Virtualized Wireless Sensor Networks' IaaS Ready for PaaS / Distributed Computing in Sensor Systems (DCOSS), 2015 International Conference.
3. Atrayee Gupta and Nandini Mukherjee Implementation of Virtual Sensors for Building a Sensor-Cloud Environment Accepted in the 8th International Conference on Communication Systems and Networks (COMSNETS) 2016.
4. M. Yuriyama, T. Kushida, and M. Itakura, A new model of accelerating service innovation with sensor-cloud infrastructure, in Proceedings of the annual SRII Global Conference (SRII '11), pp. 308–314, 2011.
5. Mihui Kim, Mihir Asthana, Siddhartha Bhargava, Kartik Krishnan Iyyer, Rohan Tangadpalliwar, Jerry Gao, Developing an On-Demand Cloud-Based Sensing-as-a-Service

System for Internet of Things // Journal of Computer Networks and Communications, Volume 2016, Article ID 3292783, 17 pages.
6. Гайдамако В.В. Инфраструктура SENSOR-CLOUD – облачные информационно-измерительные системы / Проблемы автоматики и управления. – 2018. – №2 (35). С. 109–118.
7. The Home of Location Technology Innovation and Collaboration. URL: https://www.ogc.org/, (дата обращения 08.06.2020).
8. ISO. Международная Организация по Стандартизации. URL: https://www.iso.ru/, (дата обращения 08.06.2020).
9. https://developer.mozilla.org/ru/docs/Web/XML/ (дата обращения 11.05.2020).
10. Авельцов Д.О. Применение микросервисной архитектуры в разработке программного обеспечения системы мониторинга параметров окружающей среды // Проблемы автоматики и управления. – 2018. – №1 (34). – С. 34–43.
11. Wilde E., Pautasso C. REST: From Research to Practice // Springer Science & Business Media, 2011.
12. URL:https://developer.mozilla.org/ru/docs/Web/HTTP. (дата обращения 11.05.2020)
13. URL:https://www.json.org/ (дата обращения 11.05.2020).
14. URL:52north.github.io/smle/master (дата обращения 11.05.2020).

КЫСКАЧА МАЗМУНУ

ДИНАМИКАЛЫК ТУТУМДАРДЫ ЖАНА ЖАРАЯНДАРДЫ БАШКАРУУ ЖАНА МОДЕЛДӨӨ

озгочолукторун эске алуу менен диференциалдык тендемелер системасы турундо корсотулгон математикалык модели жазылды жана талданды. Токтотмо тузулуштун математикалык моделинин негизинде, ар турдуу башкаруу схемаларында, анын ичинде токтотмонун сурулгуч туйунунун ажыратуусун кучотуу менен да, Simulink чойросундо имитациялоочу моделдери тургузулду. Имитациялык моделдерди колдонуп токтомо тузулуштун чондуктарынын озгорушуно жараша отмо жараяндары изилденилди жана ылайыктуу болгон кучотуу схемаларын тандоо боюнча сунуштар берилди.

***Негизги создор*:** токтотмо тузулуш, кучотуу схемалары, отмо жараян, моделдоо, математикалык модел

БИОЭНЕРГЕТИКАЛЫК КОМПЛЕКСТЕГИ ЭЛЕКТРДИК БАШКАРУУ СИСТЕМАСЫ / ***И.В. Бочкарев, А.Р. Сандыбаева, Х.Г. Багиев***

Ар кандай органикалык калдыктардан алардан пайдалуу заттарды алуу үчүн кайра иштетүүнүн максатка ылайыктуулугу көрсөтүлгөн. Биоэнергетикалык комплекстин структурасы жана аны иштетүү учурунда технологиялык иштердин ырааттуулугу каралат. Комплекстүү иштин эң биринчи баскычы - кыкты тазалоо процедурасы корсотулгон. Бул процедураны жүзөгө ашыруу үчүн, кык тазалоонун механикалык ыкмасы кыртышты топуруктуу кык жыйноо тутуму менен колдонулаарын, ал эки конвейерден турарын - горизонталдуу жана ийилген, ар бири өзүнүн кыймылдаткыч моторуна ээ жана бири-бирине көз каранды эмес экендиги суроттолгон. Кык чогултуу конвейерлеринин ишин аларды башкаруунун схемасын иштеп чыгууда эске алынышы керек болгон бир нече өзгөчө белгилер менен мүнөздөлгөн. Конвейерлердин жыштыгын башкаруу схемасы иштелип чыккан, анын негизи программалоочу логикалык башкаруучу болуп саналат.

***Негизги сөздөр*:** биоэнергетикалык комплекс, биореактор, кыкты тазалоо тутуму, кыргыч конвейер, конвейерди башкаруу схемасы, программаланган логикалык башкаруучу, жыштык озгорткуч.

БАРДЫК САН ШАКТАРЫНЫН ЖАРЫЯЛАНГАН ЖОГОРУ КОМПАНИЯЛАРЫ / ***А.Т. Нуртазин, З.Г. Хисамиев.***

Бул эмгекте бүтүн сандар шакектеринин компаньондорунун моделдери изилденген, бул изилдөө А.Нуртазин тарабынан иштелип чыккан Фриз класс теориясынын жардамы менен классикалык объектти изилдөөнүн мисалы болуп саналат. Изилдөөдө бүтүн сандар шакектеринин шериктери жана учурдагы жабык шериктери жөнүндө айтылат.

***Негизги создор*:** шерик; болжолдуу жабык шакек; шакек изоморфизм; алгебралык элемент; трансценденталдык элемент.

МААЛЫМАТ ТЕХНОЛОГИЯЛАРЫ ЖАНА МААЛЫМАТТАРДЫ ИШТЕП ЧЫГУУ

КРОССПАЛАТФОРМАНЫ ТРАССОИСКАТЕЛТЫ ӨНҮГҮҮ ПРОГРАММА БАШКАРУУЧУ БАГЫТЫ / ***S.N. Verzunov.***

Бул макалада локатордун кайчылаш платформа программасынын курамдык бөлүгүнүн архитектурасы сунушталат, ал максаттуу платформадан коддон бөлүнүп чыгууга негизделет, аны каалаган максаттуу платформада эч кандай өзгөртүүсүз ишке киргизүүгө болот. Кайчылаш платформалык конфигурацияга Kivy сыяктуу куралдар топтомунун жардамы менен жетишилген жана CMake, Swig жана Buildozer тиркемелери үчүн шаймандарды тапкан программанын курамдык бөлүгүн Android мобилдик тутумуна көчүрүп, Windows жана Linux иштөө тутумдарында иштөөгө мүмкүнчүлүк берген. Бул сизге мүмкүн болгон колдонуучулардын чөйрөсүн көбөйтүүгө, GPS кабыл алгычты жана көптөгөн мобилдик түзмөктөргө орнотулган

магниттик компасты колдонуу менен көзөмөлдөөчүнүн функционалдык мүмкүнчүлүгүн кеңейтүүгө мүмкүндүк берет, бул анын иш жүзүндө колдонуунун ыңгайлуулугун дагы жогорулатат.

***Негизги создор*:** кайчылаш платформа архитектурасы, Киви, buildozer, Android, Swig, LCARD E502, ARM, android and python, Docker.

КОНЦЕПЦИЯНЫ ИШТЕП ЧЫГУУ, ТҮЗҮҮ ПРОГРАММАЛЫК-АППАРАТТЫК ӨЗӨК УНИВЕРСАЛДУУ ЧӨЙРӨНҮ ДОЛБООРЛООНУН АВТОМАТТАШТЫРЫЛГАН СИСТЕМАЛАРДЫ КОРГОЛГОН АТКАРУУ / ***С.В. Корякин.***

Беренесинде маселелер каралат куруу программалык-аппараттык өзөк универсалдуу чөйрөнү долбоорлоонун автоматташтырылган системаларды корголгон аткаруу(АСЗИ) пайдалануу менен бириктирилип системасынын айкын убакыт ыргагында(СРВ). Чөйрө түзүлгөн системаларды тез моделдөө жана алгоритмдерди иштетүү мүмкүндүгүн камсыз кылуу менен иштеп чыгуунун толук циклин колдойт. Инвестиция агенттиги тарабынан иштелип чыккан универсалдуу долбоорлоо чөйрөсүн куруунун сыпаттамасы, концепциясы келтирилген.

Негизги сөздөр: Модели; долбоорлоо этаптары; долбоорлоо чөйрөсү; реалдуу убакыт системасы (РУС); корголгон аткаруунун автоматташтырылган системасы.

УБАКТЫН САПАТЫ ИНДЕКСИН КЛАССИФИКАЦИЯЛОО ҮЧҮН LSTM-NEURAL ТАРМАКТАРЫН ӨТҮНҮҮ БИШКЕК ШААРЫ / ***Н.М Лыченко., А.В. Сороковая.***

Нейрондук тармакты классификациялоо милдети катары метеорологиялык параметрлерге жараша Бишкектеги абанын сапатынын индексин болжолдоо маселеси каралат. LSTM нейрон тармагын эң натыйжалуу деп тандоо негизделген. Абанын сапатынын индексинин классификатору иштелип чыккан, ал метеорологиялык параметрлерди жана ар кандай болжолдуу тереңдиктерди байкоонун ар кандай тарыхы үчүн "Жакшы" / "Ден-соолукка зыяндуу" деп болжолдоо көйгөйүн чечет. Болжолдуу 90% дан ашык тактыкка жетишилди.

Ачкыч сөздөр: классификация, болжолдоо, абанын сапатынын индекси, LSTM-нейрон тармагы.

SQL ТАПШЫРМАЛАРЫНЫН ЧЕЧИМДЕРИН ТЕКШЕРҮҮ ИШТЕРИН АВТОМАТТАШТЫРУУ/ *Р. Ш. Мустафин О. М.С. Осмонов.*

Туура SQL сурамдарын түзүү жөндөмү - бул көптөгөн программалык камсыздоону иштеп чыгуучуларга керек болгон негизги өнөрдүлүк.

Бул жөндөмдү өздөштүрүү бир топ күч-аракетти жана практикалык чеберчиликти талап кылган татаал процесс. Сурамдарды туура түзү көндүмдөрүн текшерүү процессин автоматташтыруусу ушундай окутуунун натыйжалуулугун кыйла жогорулатат. Сунушталган бир катар чечимдерге карабастан, азыркы учурда билим текшерүнүн автоматташтырылган тутумдарын түзүү маселесин чечүүдө бирдиктүү мамилелер жок, ошондой эле алардын программалык камсыздоосун жүзөгө ашырууга коюлган талаптарда жок.

Бул макалада DML (Data Manipulation Language) көрсөтмөлөрүн колдонуунун негизинде SQL сурамдарынын натыйжаларын текшерүүнү автоматташтыруу ыкмасы сунуш кылынган, ар кандай жүзөгө ашырылган маалыматтар базасын башкаруу тутумдарында бир тапшырманын чечилишин текшерүү мүмкүнчүлүгү менен.

Негизги сөздөр: SQL, электрондук окутуу, тапшырмаларды чечүү, маалыматтар базасын башкаруу тутуму, маалымат базасы, автоматташтыруу.

БӨЛҮШТҮРҮЛГӨН МААЛЫМАТТЫК-ЧЕНӨӨЧҮ СИСТЕМАЛАР ҮЧҮН СЕНСОРДУК ТҮЗҮЛҮШТӨРДҮ ВИРТУАЛДАШТЫРУУ МОДУЛУН ИШТЕП ЧЫГУУ / ***Д.Авельцов, В. Гайдамако.***

Бул эмгекте ченөөчү түзүлүштөрдү кызмат катары берүүчү, булуттуу ченөө системаларында бөлүштүрүлгөн буюмдарды Интернетте пайдалануу үчүн сенсордук виртуалдаштыруу модулун иштеп чыгуунун мисалы каралды. Системаны ишке ашырууда OGC датчиктерин сүрөттөө стандарттары пайдаланылды. Ошондой эле стандарттардын маалыматтарынын негизинде моделдердин маалыматтары жана алардын ортосундагы байланыш имплементацияланды. Колдонуучу менен өз ара аракеттенишүү модулу иштелип чыкты жана тестирлөө жүргүзүлдү.
Негизги сөздөр: сенсордук түзмөктөр, виртуалдаштыруу, билгизгичтерди виртуалдаштыруу, docker, docker-контейнери, docker тармактары, docker cli , микросервистер (microservices), OGC (Open Geospatial Cons4ortium), sensorML, буюмдарды Интернети, маалымат өлчөө тутумдары (МӨТ)

ABSTRAKT

MANAGEMENT AND MODELING OF DYNAMIC SYSTEMS AND PROCESSES

ON APPROACHES OF SIMPLIFICATION OF COMPLEX DYNAMIC SYSTEMS / ***Zh. Sharshenaliev.***

Methods for the study of complex objects, a systematic approach, interaction are described.
Keywords: general theory of systems, systems approach; decomposition; aggregation; feedback; mutual cooperation

FEEDBACK OPTIMUM CONTROL ALGORITHM FOR THE PROCESS WITH THE CONSEQUENCE / ***T.P. Samokhvalova.***

A feedback control algorithm is constructed by the R. Bellman method for a one-dimensional process with aftereffect.
Keywords*:* optimal control; feedback; the optimality principle of R. Bellman; Volterr type integro-differential equation; derivative in the direction.

METHOD OF RELIABILITY ANALYSIS AND DIAGNOSTICS OF SMALL HYDRO-ELECTRIC STATION FOR AUTONOMOUS POWER SUPPLY / ***A.B.Bakasova, K.Satarkulov, G.N.Niiazova, T.K.Satarkulov.***

The article provides a brief overview of small hydro-electric station with a centrifugal regulator and a flywheel with automatically controlled mass and moment of inertia. The method of reliability analysis and diagnostics of states was described. Small hydro-electric station was represented as an *S* physical system with a Markov random process with discrete states and continuous time. It is assumed that all transitions of *S* system from one state to another occur under the action of the simplest flows of failure λ and recovery μ with the specified intensities. A marked column of states of the system is compiled and a mathematical model of the considered process is constructed, which makes it possible to find all the probabilities of states as a function of time, i.e. to diagnose states. Besides, the resulting mathematical model allows to evaluate the operation of a small hydro-electric station taking into account all the conditions on which its functionality depends, and to assess the losses caused by various reasons.
Keywords: small hydro-electric station, flywheel, rotation frequency stabilization, hydraulic turbine, Watt's governor, mathematical model, Markov process, marked column, reliability analysis, diagnosis of states.

RESEARCH OF TRANSITION PROCESSES IN AN ELECTROMECHANICAL BRAKE DEVICE TAKING INTO ACCOUNT THE DYNAMICS OF ANCHOR MOTION / *I.V. Bochkarev, D.N Sadykov*

The article studies transients occurring in electromechanical braking devices, depending on their control scheme. The laws of changing the current and flux linkage of an electromagnet in various operating modes are considered. A mathematical model of the braking device is compiled and analyzed in the form of a system of differential equations, taking into account the features of its design and operating modes. Based on the mathematical model, simulation models of the brake device in the Simulink environment were built with various control schemes, including with the force of opening the brake friction unit. Using simulation models, transient studies in the braking device were studied depending on changes in its parameters, and recommendations were made on choosing the most appropriate boosting schemes.

Keywords: braking device, boost circuit, transient, modeling, mathematical model

ELECTRIC CONTROL SYSTEM BIOENERGY COMPLEX / *I.V. Bochkarev, A.R. Sandybaeva, H. G. Bagiev.*

The expediency of processing various organic wastes in order to obtain useful substances from them is shown. The structure of the bioenergy complex and the sequence of technological operations during its operation are considered. The very first stage of the complex operation is described - the manure removal procedure. It is shown that for the implementation of this procedure, a mechanical method of manure removal is used by means of a system of scraper manure collection conveyors, which consists of two conveyors - horizontal and inclined, each of which has its own drive motor and operates independently of each other. It is shown that the work of manure collection conveyors is characterized by a number of specific features that must be taken into account when developing a scheme for their management. A scheme for the frequency control of conveyors is developed, the basis of which is a programmable logic controller.

Keywords*:* bioenergy complex, bioreactor, manure removal system, scraper conveyor, conveyor control circuit, programmable logic controller, frequency converter.

EXISTENTIALLY CLOSED COMPANIONS OF WHOLE NUMBER RINGS / *Nurtazin A.T., Khisamiev Z.G.*

Companion models of the ring of integers are studied in this work. This study is an example of the study of a classical object by means of the Frese class theory developed by A. Nurtazin. The study describes the structure of companions and existentially closed companions of the ring of integers.

Keywords: companion; existentially closed ring; ring isomorphism; algebraic element; transcendental element.

INFORMATION TECHNOLOGIES AND INFORMATION PROCESSING

CROSSPLATFORM DEVELOPMENT TRACE RECORDER SOFTWARE COMPONENT / *S. N. Verzunov*,

This article proposes the architecture of a cross-platform software component of the locator, based on the separation of code that depends on the target platform from the code, which can be launched on any target platform without any changes. Cross-platform configuration was achieved with the help of toolkits such as Kivy and build tools for CMake, Swig and Buildozer applications, which allow porting the tracer-finder software component to the Android mobile operating system, while keeping it functional on Windows and Linux desktop operating systems. This allows you to increase the potential circle of possible users, and expand the functionality of the tracer through the use of a GPS receiver and a magnetic compass built into many mobile devices, which also increases the convenience of its practical use.

***Keywords**:* cross-platform architecture, Kivy, buildozer, Android, Swig, LCARD E502, ARM, python-for-android, Docker.

DEVELOPMENT OF THE CONCEPT OF BUILDING THE HARDWARE AND SOFTWARE CORE OF THE UNIVERSAL DESIGN ENVIRONMENT FOR AUTOMATED SYSTEMS OF PROTECTED EXECUTION // *S. V. Koryakin,*

The article deals with the issues of building the hardware and software core of the universal design environment for automated systems of protected execution (ASSI) using embedded real -time systems (SRS). The environment supports a full development cycle, providing high- speed simulation of the created systems and processing algorithms. The description of the concept of building the developed universal design environment of the ASSI is given.

Keywords: Model; design stages; design environment; real-time systems (SRV); automated system of secure execution.

LSTM-NEURAL NETWORKS FOR CLASSIFICATION OF THE AIR QUALITY INDEX OF BISHKEK CITY / ***N.M. Lychenko, A.V. Sorokovaya***

The problem of forecasting the air quality index AQI in Bishkek, depending on meteorological parameters, is considered as a task of neural network classification. The choice of the LSTM neural network as the most effective is justified. A classifier of the air quality index has been developed to solve the problem of forecasting the AQI classes “Good” / “Unhealthy” for a different history of observations of meteorological parameters and different forecast depths. A forecast accuracy of more than 90% has been achieved.

***Keywords**:* classification, forecast, air quality index, LSTM-neural network.

AUTOMATION OF CHECKING SQL TASK SOLUTIONS / ***R.Sh.Mustafin, M.S.Osmonov.***

The ability to create correct SQL queries is a fundamental skill that many software developers need. Mastering this skill is a complex process that requires considerable effort and practice. Automating the process of checking skills for correct query writing significantly improves the effectiveness of such training. Despite a number of proposed solutions, there are currently no unified approaches to solving the problem of building automated knowledge control systems, as well as requirements for their software implementation.

This article offers an approach to automate checking the result of SQL queries based on the use of DML (Data Manipulation Language) instructions, with the opportunity to check the solution of the same task on different DBMS implementations.

Keywords: SQL, e-learning, task solving, DBMS, database, automation

DEVELOPMENT OF A SENSOR DEVICES VIRTUALIZATION MODULE FOR DISTRIBUTED INFORMATION-MEASURING SYSTEMS / ***D.Aveltsov, V.Gaidamako***

In this paper, an example of the development of a virtualization module for sensor devices used in the Internet of things, distributed and cloud information-measuring systems that provide measuring devices as a service is considered. When implementing the system, OGC sensor description standards were used. Also based on these standards, data models and the relationships between them were implemented. A user interaction module has been developed and tested.

Keywords: sensor devices, virtualization, sensors virtualization, docker, docker-container, docker networks, docker cli, microservices, OGC (Open Geospatial Consortium), sensorML, Internet of Things, information-measuring systems.

ПРОБЛЕМЫ АВТОМАТИКИ И УПРАВЛЕНИЯ

Научно-технический журнал

Компьютерная верстка В.П. Алексеева

Подписано к печати 15.06.2020 г. Формат 70/108 1/8
Печать офсетная. Объем 12,1 п.л. Тираж 200 экз.

Издательство «Илим»
720071, Бишкек, проспект Чуй, 265-а