МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАТИКА И КИБЕРНЕТИКА

1 (35)

Донецк – 2024

УДК 004.3+004.9+004.2+51.7+519.6+519.7

ИНФОРМАТИКА И КИБЕРНЕТИКА, № 1 (35), 2024,
Донецк, ДонНТУ.

Выпуск подготовлен по материалам, подготовленным для обсуждения на XV Международной научно-технической конференции «Информатика, управляющие системы, математическое и компьютерное моделирование – 2024» (ИУСМКМ–2024), дата проведения 29–30 мая 2024 г. в рамках X Научного форума Донецкой Народной Республики, а также результатам текущей научно-технической деятельности аспирантов, соискателей и научных работников. Статьи посвящены вопросам приоритетных направлений в области информатики, кибернетики, вычислительной техники и интеллектуальных систем.

Материалы предназначены для специалистов народного хозяйства, ученых, преподавателей, аспирантов и студентов высших учебных заведений.

Рекомендовано к печати ученым советом ФГБОУ ВО «Донецкий национальный технический университет» Министерства науки и высшего образования РФ. Протокол № 3 от 24 апреля 2024 г.

Свидетельство о регистрации СМИ: серия ААА № 000145 от 20.06.2017.

Приказ МОН ДНР № 135 от 01.02.2019 о включении в Перечень рецензируемых научных изданий ВАК ДНР.

Контактный адрес редакции
РФ, ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36., ул. Кобозева, 17.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
Эл. почта: infcyb.donntu@yandex.ru
Интернет: http://infcyb.donntu.ru

СОДЕРЖАНИЕ

Информатика и вычислительная техника

Информатика и вычислительная техника

УДК 004.8

Моделирование и прогнозирование потребления тепловой энергии с использованием погодных данных

А. Б. Бирюков, С. В. Гридин
Донецкий национальный технический университет, г. Донецк

e-mail: birukov.donntu@mail.ru

Аннотация
Работа посвящена моделированию и прогнозированию потребления тепловой энергии объектом сферы здравоохранения в зависимости от данных о погодных условиях на основе алгоритмов машинного обучения. В качестве алгоритмов были использованы Random Forest и Support Vector Machines, так как именно эти алгоритмы более всего подходят для решения задач регрессии. Точность разработанных моделей проверялась и сопоставлялась на тестовых данных. Модели, разработанные на основе указанных алгоритмов, показывают достаточно высокую эффективность в условиях ограниченного количества данных для обучения и имеют значительный потенциал для использования.

Введение

Как для отдельно взятого потребителя, так и для системы «генерация-транспортировка-потребление» тепловой энергии в целом, основным фактором, определяющим уровень потребления, являются погодные условия. Техническое состояние потребителя и систем теплоснабжения, эффективность основного и вспомогательного оборудования объектов генерации, степень автоматизации процессов регулирования и распределения теплоносителя влияют на уровень тепловых потерь и класс энергетической эффективности, саму же тепловую нагрузку определяет состояние окружающей среды.

На потребление объектом тепловой энергии на протяжении отопительного сезона влияет не только непосредственно температура наружного воздуха. Достаточно существенными факторами могут быть сила и направление ветра, а также интенсивность солнечного излучения. Сила ветра оказывает влияние на теплопотери с инфильтрацией. В свою очередь, чем интенсивнее солнечное излучение, тем выше теплопоступления. При этом степень влияния ветра и солнечного излучения в значительной степени зависят от конструкционных особенностей здания-потребителя: расположение здания в пространстве, высота, площадь и ориентированность светопрозрачных ограждающих конструкций, уровень герметичности наружных стен и т.п. [1]. В связи с этим, влияние этих факторов необходимо оценивать индивидуально для каждого объекта.

На потребление тепловой энергии может оказать влияние не только среднесуточная температура воздуха, но и температурный перепад, т.к. при резком повышении или снижении температуры воздуха могут возникать кратковременные перепады тепловой нагрузки, величина которых зависит от инерционности и уровня автоматизации системы генерации.

Имея в своем распоряжении погодные данные за определенный период, а также данные по потреблению тепловой энергии (к примеру, показания теплового счетчика) или непосредственно потребления первичного энергоресурса (природного газа, электроэнергии), можно построить модель потребления энергии на нужды отопления для конкретного объекта. Такая модель позволяет определять и прогнозировать потребление тепловой энергии объектом в зависимости от погодных факторов (температура и температурный перепад, сила ветра, солнечное излучение, влажность, осадки) с учетом их значимости [1].

Постановка задачи моделирования потребления тепловой энергии в зависимости от погодных характеристик

Моделирование и прогнозирование потребления тепловой энергии позволяют решить множество задач на уровне потребителя тепловой энергии, поставщика тепловой энергии и системы энергетического менеджмента, таких как:

- планирование энергетического бюджета. Имея в своем распоряжении достаточно точные погодные данные, можно спрогнозировать затраты на тепловую энергию для определенного периода;
- выявление снижения энергоэффективности объекта. Сопоставляя данные по потреблению тепловой энергии, полученные с помощью модели, с фактическими данными по потреблению (данные узла учета

тепловой энергии), можно выявить отклонения, которые могут свидетельствовать о нарушениях в работе системы отопления объекта. Выявив эти нарушения, потребитель или обслуживающая организация могут их своевременно устранить или запланировать мероприятия по повышению энергоэффективности объекта;

- внедрение моделей потребления в локальные системы погодозависимого регулирования тепловой нагрузки для поддержания параметров теплового комфорта на необходимом уровне независимо от колебаний погодных условий;

- использование моделей потребления для снижения инерционности объектов генерации тепловой энергии за счет прогнозирования резких изменений погодных условий и, соответственно, тепловой нагрузки;

- использование моделей потребления для эффективного распределения энергетических потоков. При совместном использовании традиционных и альтернативных источников тепловой энергии существует необходимость поддержания наиболее эффективного баланса в системе потребления: при соблюдении графика тепловой нагрузки необходимо поддерживать максимально возможную долю энергии от альтернативных источников. Точное прогнозирование потребления тепловой энергии позволяет эффективно регулировать поступление энергии от различных источников.

С учетом вышесказанного, целью статьи является построение модели прогнозирования потребления тепловой энергии с использованием методов машинного обучения (Machine Learning, ML) на примере конкретного объекта, с использованием фактических погодных данных.

Объект исследования

Объектом исследования является комплекс блоков учреждения здравоохранения, расположенный в г. Донецк Донецкой Народной Республики. Здания комплекса 3-х и 4-х этажные, стены выполнены из силикатного кирпича толщиной не менее 0,5 м. Дополнительная теплоизоляция стен отсутствует. В зданиях присутствует техническое подполье и подвальные помещения с расположенными коммуникациями системы теплоснабжения, водоснабжения и канализации, а также неотапливаемые чердачные помещения. Окна в здании частично пластиковые, частично старые деревянные. Пластиковые окна низкоэффективные, однокамерные, без низкоэмиссионного покрытия. Часть старых деревянных окон находится в неудовлетворительном состоянии, что обуславливает увеличение тепловых потерь с инфильтрацией.

Трубопроводы системы отопления здания с нижней разводкой, теплоизоляция трубопроводов в неотапливаемых помещениях либо отсутствует, либо находится в неудовлетворительном состоянии. Система отопления несбалансированная. Радиаторы чугунные и стальные, термостаты и оборудование контроля температурного режима в помещениях отсутствует. Узлы учета тепловой энергии присутствуют, учет не автоматизированный, показания фиксируются вручную.

Теплоснабжение комплекса объекта здравоохранения осуществляется за счет собственной котельной, находящейся непосредственно на территории комплекса. В котельной установлены 3 одноконтурных водогрейных газовых котла с КПД не ниже 0,9. На котлах установлены автоматизированные панели управления, осуществляющие регулировку работы горелки по заданной температуре теплоносителя. Вмешательство персонала котельной в работу котла минимально. Котельная обслуживает исключительно рассматриваемый объект сферы здравоохранения. Фиксация расхода природного газа осуществляется за счет установленного в котельной счетчика природного газа и корректора расхода газа. Съём показаний корректора расхода газа осуществляется как в ручном, так и в автоматизированном режиме.

Теплотрассы системы теплоснабжения подземной укладки в каналах, теплоизоляция присутствует. Все теплотрассы располагаются непосредственно на территории рассматриваемого объекта.

Данные о потреблении объектом энергетических ресурсов

На данный момент сбор данных о потреблении тем или иным объектом энергетических ресурсов может быть затруднителен. Далеко не все объекты теплоснабжения, особенно в жилищно-коммунальном секторе, оборудованы узлами учета тепловой энергии. Сбор данных не автоматизирован, оборудование для съема показаний на электронные носители отсутствует. Ручной сбор данных может носить несистемный характер и иметь значительную погрешность. В то же время, для проведения исследований, связанных с анализом потребления энергоресурсов, нужны надежные и стабильные данные для рассматриваемого временного интервала.

Для исследуемого объекта имеются частичные данные суточного потребления тепловой энергии за период с 01.01.2018 г. по 02.01.2019 г., а также с 11.12.2019 г. по 22.04.2020 г (рис. 1, рис. 2).

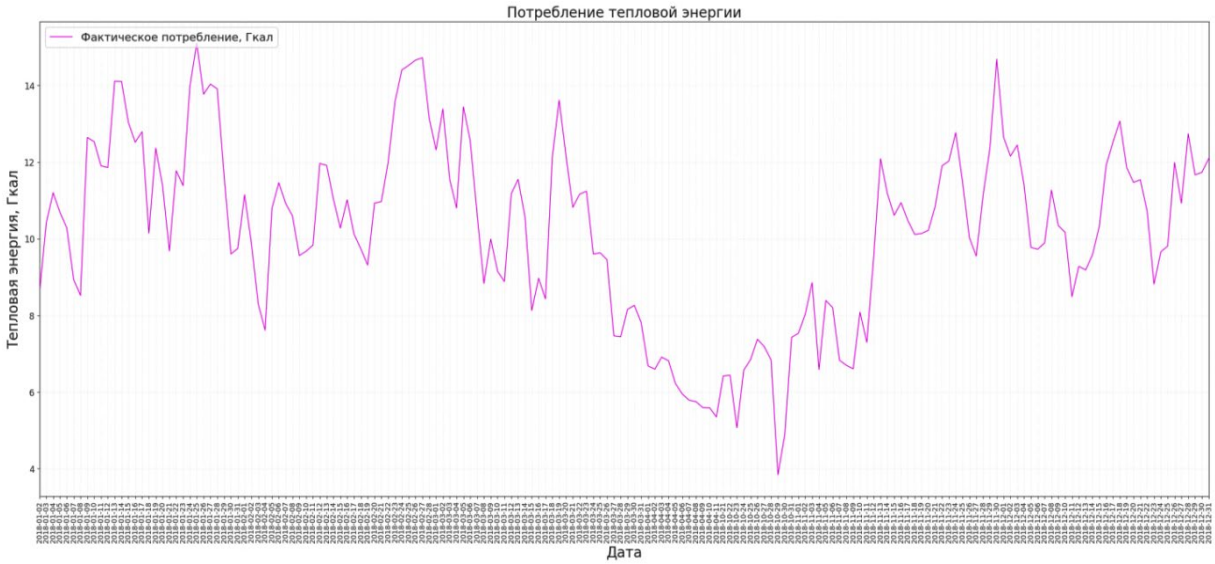


Рисунок 1 – Фактическое потребление тепловой энергии в 2018 г.

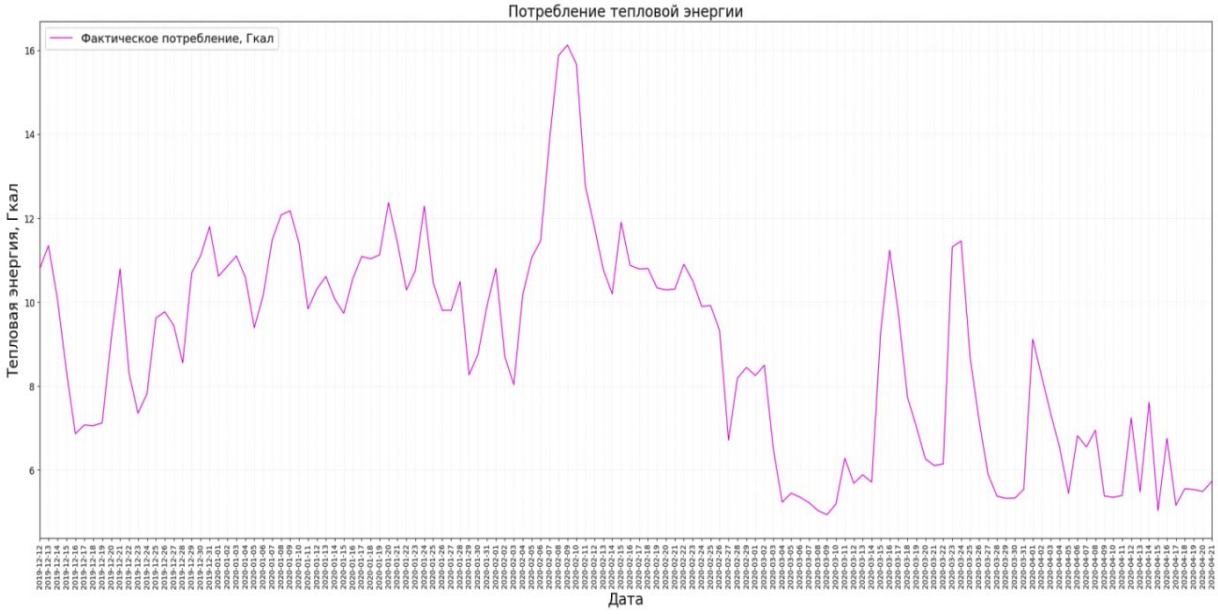


Рисунок 2 – Фактическое потребление тепловой энергии в 2019-2020 г.г.

Несмотря на то, что данные о потреблении тепловой энергии ограничены, тем не менее, для построения достаточно эффективной модели прогнозирования с использованием современных методов машинного обучения их может быть достаточно, как будет показано далее. При дальнейшем сборе данных и использовании их для обучения разрабатываемой модели эффективность и точность модели будет только возрастать.

Из используемых в исследовании данных потребления тепловой энергии были выделены для тестирования модели данные за февраль – март 2020 г. Остальные данные использовались для обучения регрессионной модели.

Погодные данные для анализа

Для исследования были использованы фактические погодные данные для г. Донецк за исследуемый период. Фрагмент полного набора данных отображен на рис. 3. Набор данных, используемый для обучения модели, отображен на рис. 4.

	name	datetime	tempmax	tempmin	temp	feelslikemax	feelslikemin	feelslike	dew	humidity	...	solarenergy	uvindex	severerisk	sunrise	sunset	moonphase	conditions	description	icon
0	Donetsk	2018-01-02	4.8	0.5	2.6	1.5	-4.6	-0.9	1.8	94.5	...	3.4	2	NaN	02.01.2018 7:18	02.01.2018 15:46	0.50	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day
1	Donetsk	2018-01-03	2.6	-2.7	-0.4	-2.8	-8.7	-6.3	-2.1	89.1	...	3.9	3	NaN	03.01.2018 7:18	03.01.2018 15:47	0.54	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day
2	Donetsk	2018-01-04	1.3	-3.1	-0.9	-3.9	-9.3	-6.5	-2.4	89.7	...	2.6	2	NaN	04.01.2018 7:18	04.01.2018 15:49	0.58	Snow, Rain, Overcast	Cloudy skies throughout the day with rain or s...	rain
3	Donetsk	2018-01-05	2.7	-1.5	0.0	-0.6	-6.5	-4.3	-1.9	87.3	...	3.9	2	NaN	05.01.2018 7:18	05.01.2018 15:50	0.61	Snow, Rain, Partially cloudy	Partly cloudy throughout the day with early mo...	rain
4	Donetsk	2018-01-06	3.9	-0.9	1.0	3.4	-3.3	-0.9	-0.2	91.4	...	3.1	2	NaN	06.01.2018 7:18	06.01.2018 15:51	0.64	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
300	Donetsk	2020-04-17	18.4	6.8	11.7	18.4	2.0	9.9	4.5	64.3	...	18.5	7	NaN	17.04.2020 5:35	17.04.2020 19:22	0.81	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day
301	Donetsk	2020-04-18	17.2	5.6	10.8	17.2	3.2	10.1	0.6	53.2	...	23.0	8	NaN	18.04.2020 5:33	18.04.2020 19:23	0.84	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day
302	Donetsk	2020-04-19	15.2	3.8	9.8	15.2	2.0	9.0	0.2	55.5	...	15.6	5	NaN	19.04.2020 5:31	19.04.2020 19:25	0.88	Rain, Partially cloudy	Partly cloudy throughout the day with late aft...	rain
303	Donetsk	2020-04-20	13.0	3.1	7.6	13.0	-0.8	6.0	1.9	71.0	...	15.5	7	NaN	20.04.2020 5:29	20.04.2020 19:26	0.91	Rain, Partially cloudy	Clearing in the afternoon with a chance of rai...	rain
304	Donetsk	2020-04-21	10.1	0.1	5.6	10.1	-3.9	2.4	-0.5	68.0	...	17.4	6	NaN	21.04.2020 5:27	21.04.2020 19:28	0.94	Partially cloudy	Partly cloudy throughout the day.	partly-cloudy-day

305 rows × 33 columns

Рисунок 3 – Погодные данные для г. Донецка

	date	temp	tempmin	tempmax	humidity	precip	windspeed	cloudcover	solarenergy
0	2018-01-02	2.6	0.5	4.8	94.5	0.0	19.4	75.1	3.4
1	2018-01-03	-0.4	-2.7	2.6	89.1	0.0	29.2	69.6	3.9
2	2018-01-04	-0.9	-3.1	1.3	89.7	3.3	24.8	98.1	2.6
3	2018-01-05	0.0	-1.5	2.7	87.3	0.2	18.7	83.0	3.9
4	2018-01-06	1.0	-0.9	3.9	91.4	0.0	16.2	84.6	3.1
...	...	...	...	...	...	...	...	...	...
300	2020-04-17	11.7	6.8	18.4	64.3	0.0	38.5	81.5	18.5
301	2020-04-18	10.8	5.6	17.2	53.2	0.0	23.4	50.3	23.0
302	2020-04-19	9.8	3.8	15.2	55.5	0.8	14.4	87.8	15.6
303	2020-04-20	7.6	3.1	13.0	71.0	13.2	24.8	57.5	15.5
304	2020-04-21	5.6	0.1	10.1	68.0	0.0	22.3	25.8	17.4

305 rows × 9 columns

Рисунок 4 – Погодные данные, используемые для обучения модели

При разработке модели были использованы следующие данные (рис. 4): temp – среднесуточная температура воздуха; tempmin – минимальная температура за рассматриваемые сутки; tempmax – максимальная температура для рассматриваемых суток; humidity – относительная влажность воздуха; precip – количество выпавших осадков; windspeed – средняя скорость ветра; cloudcover – облачность, solarenergy – солнечная энергия. Основополагающим фактором, влияющим на тепловую нагрузку, является среднесуточная температура, однако суточный температурный перепад (разница между значениями максимальной и минимальной температурами воздуха) также оказывает влияние на потребление тепловой энергии, т.к. с ним связано изменение режимов работы объекта тепловой генерации (котельной установки). Теплопоступления от солнечного излучения являются одним из факторов, определяющих расчетную тепловую нагрузку объекта, как и

сила и повторяемость ветра, влияющие на теплопотери с инфильтрацией. Относительная влажность и осадки являются, скорее, косвенными факторами, однако также были использованы для построения модели. Следует отметить, что указанные параметры могут влиять на ощущение теплового комфорта и для помещений с возможностью контроля параметров микроклимата могут способствовать увеличения тепловой нагрузки.

Используемые инструменты данные

При проведении исследования использовался язык программирования Python совместно с рядом библиотек: sklear, matplotlib, Pandas, sqlite3. Выбор Python связан с тем, что этот язык достаточно прост в изучении и обладает рядом библиотек и модулей, разработанных специально для выполнения ряда задач из области инженерного эксперимента, математического анализа, работы с данными, статистики и др. Это позволяет использовать язык для научных разработок специалистам, чей профиль непосредственно не связан с программированием.

Для первоначальной обработки данных о потреблении энергетических ресурсов и погодных данных использовался язык работы с базами данных *SQL*.

Для построения регрессионной модели прогнозирования потребления тепловой энергии рассматриваемого объекта использовались методы машинного обучения (*Random forest* и *Support Vector Machines*).

Машинное обучение (*Machine Learning*, *ML*) – совокупность методов, позволяющая создавать самообучающиеся компьютерные системы. Помимо более глобальной задачи по созданию нейросетей и искусственного интеллекта, методы машинного обучения используются для более узких задач: классификация, регрессия, кластеризация и д.р. [3-5].

Существует ряд преимуществ использования алгоритмов машинного обучения для решения задач регрессии: возможность анализировать очень большие объемы данных, возможность постоянного повышения эффективности модели за счет обновления входных данных (к примеру, погодных), высокая скорость осуществления расчетов и точность разработанных моделей.

В нашем случае, для построения модели прогнозирования потребления энергетических ресурсов, использовались наиболее подходящие алгоритмы для задач, связанных с регрессией – *Random forest* (дерево решений) и *Support Vector Machines* (метод опорных векторов) [3, 4].

Алгоритм *Random forest* достаточно точный, для построения модели с использованием библиотеки *sclearn* необходимо незначительное число параметров. В общем виде, работа алгоритма основывается на изучении иерархии «Если – то…», что в результате приводит к определенному решению [3, 4].

Алгоритм *Support Vector Machines* является наиболее универсальным и используется во многих направлениях. Работа алгоритма основывается на создании в *n*-мерном пространстве гиперплоскости, которая разделяет данные на классы. Точность предсказания зависит от расстояния от гиперплоскости до объектов – чем оно больше, тем выше точность [3, 4].

На основе используемых погодных данных и данных о потреблении тепловой энергии будет сопоставлена эффективность этих двух алгоритмов в условиях ограниченного количества данных для обучения модели.

Для определения значимости используемых для обучения модели переменных (в нашем случае – погодных данных) использовался коэффициент корреляции Пирсона. В общем виде, выражение для расчета коэффициента корреляции Пирсона имеет вид [2, 3]:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \cdot \sum_{i=1}^{n}(y_i - \bar{y})^2}}, \quad (1)$$

где x_i, y_i – числовые значения рассматриваемых переменных,

n – объем выборки.

Для анализа эффективности разработанной модели прогнозирования использовались следующие параметры: коэффициент детерминации R^2 и среднеквадратическая ошибка MSE [2, 3].

Коэффициент детерминации R^2 показывает, насколько близки результаты, спрогнозированные моделью, к реальным данным. В общем виде, выражение для определения коэффициента детерминации R^2 имеет вид:

$$R^2 = 1 - \frac{\sigma^2}{\sigma_y^2}, \quad (2)$$

где σ^2 - условная дисперсия зависимой переменной (дисперсия ошибки модели);

σ_y^2 - дисперсия случайной величины y.

Среднеквадратическая ошибка измеряет среднее значение квадратов ошибок, то есть

среднеквадратичную разницу между оценочными значениями и фактическим значением. Она имеет положительное значение, при этом, чем меньше её значение, тем более точной является рассматриваемая модель.

В общем виде выражение для определения среднеквадратической ошибки MSE имеет вид:

$$MSE = \frac{1}{n} \cdot \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \quad (2)$$

,

где n - количество наблюдений;
Y_i – фактическое значение;
$\hat{Y}_i$ - предсказанное значение.

Разработка регрессионной модели прогнозирования потребления тепловой энергии в зависимости от погодных данных

Итоговые данные для обучения модели отображены на рис. 5, где heat_cons – суточное потребление рассматриваемым объектом тепловой энергии в Гкал.

Значения коэффициента корреляции Пирсона отображены на рис. 6.

Ожидаемо, наибольшее влияние на потребление оказывают температуры наружного воздуха. Остальные факторы имеют умеренную и слабую корреляцию.

	date	temp	tempmin	tempmax	humidity	precip	windspeed	cloudcover	solarenergy	heat_cons
0	2018-01-02	2.6	0.5	4.8	94.5	0.0	19.4	75.1	3.4	8.5811
1	2018-01-03	-0.4	-2.7	2.6	89.1	0.0	29.2	69.6	3.9	10.4148
2	2018-01-04	-0.9	-3.1	1.3	89.7	3.3	24.8	98.1	2.6	11.1996
3	2018-01-05	0.0	-1.5	2.7	87.3	0.2	18.7	83.0	3.9	10.6862
4	2018-01-06	1.0	-0.9	3.9	91.4	0.0	16.2	84.6	3.1	10.2795
...	...	...	...	...	...	...	...	...	...	...
240	2020-04-17	11.7	6.8	18.4	64.3	0.0	38.5	81.5	18.5	5.1600
241	2020-04-18	10.8	5.6	17.2	53.2	0.0	23.4	50.3	23.0	5.5579
242	2020-04-19	9.8	3.8	15.2	55.5	0.8	14.4	87.8	15.6	5.5358
243	2020-04-20	7.6	3.1	13.0	71.0	13.2	24.8	57.5	15.5	5.4902
244	2020-04-21	5.6	0.1	10.1	68.0	0.0	22.3	25.8	17.4	5.7256

245 rows × 10 columns

Рисунок 5 – Входные данные для обучения модели

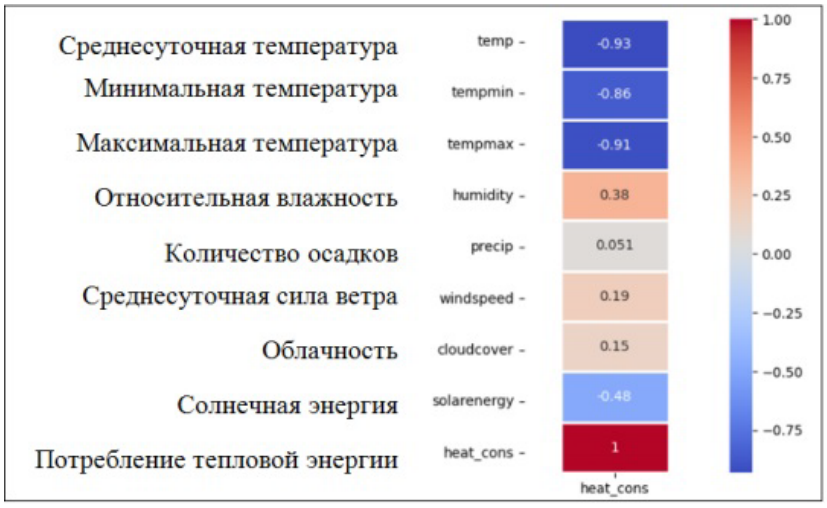


Рисунок 6 – Значения коэффициента корреляции Пирсона

Для обучения модели будут использованы все указанные погодные параметры. Обучение модели для прогнозирования потребления тепловой энергии при использовании алгоритма Random Forest.

Фрагмент кода, с помощью которого импортируется необходимый алгоритм, вносятся некоторые параметры, вносятся тренировочные данные, а также рассчитывается коэффициент детерминации R^2 и среднеквадратическая ошибка отображен на рис. 7.

```
#Импортирование библиотек
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor

#Задание параметров для модели
RFreg_daily= RandomForestRegressor(n_estimators = 500, max_features ='log2', max_depth = 7, random_state=0)
RFreg_daily.fit(x_train,y_train)
Predicted_train = RFreg_daily.predict(x_train)

#Анализ эффективности модели
print(r2_score(y_train,Predicted_train))
print(mean_squared_error(y_train,Predicted_train))

0.9733644444893533
0.14894048036277562
```

Рисунок 7 – Значения коэффициента корреляции Пирсона

Значения коэффициента детерминации и среднеквадратической ошибки для тренировочных данных отображены в табл. 1.

Таблица 1 – Критерии эффективности модели

Параметр	Значение
Коэффициент детерминации R^2	0,97
Среднеквадратическая ошибка MSE	0,15

Фактическое и прогнозируемое потребление тепловой энергии для тренировочных данных отображено на рис. 8. Исходя из данных табл. 1 и данных графика на рис. 8, разработанная модель показывает достаточную эффективность на тренировочных данных. Далее необходимо проверить модель на тестовых данных. Визуализация фрагмента дерева принятия решений созданной модели изображена на рис. 9.

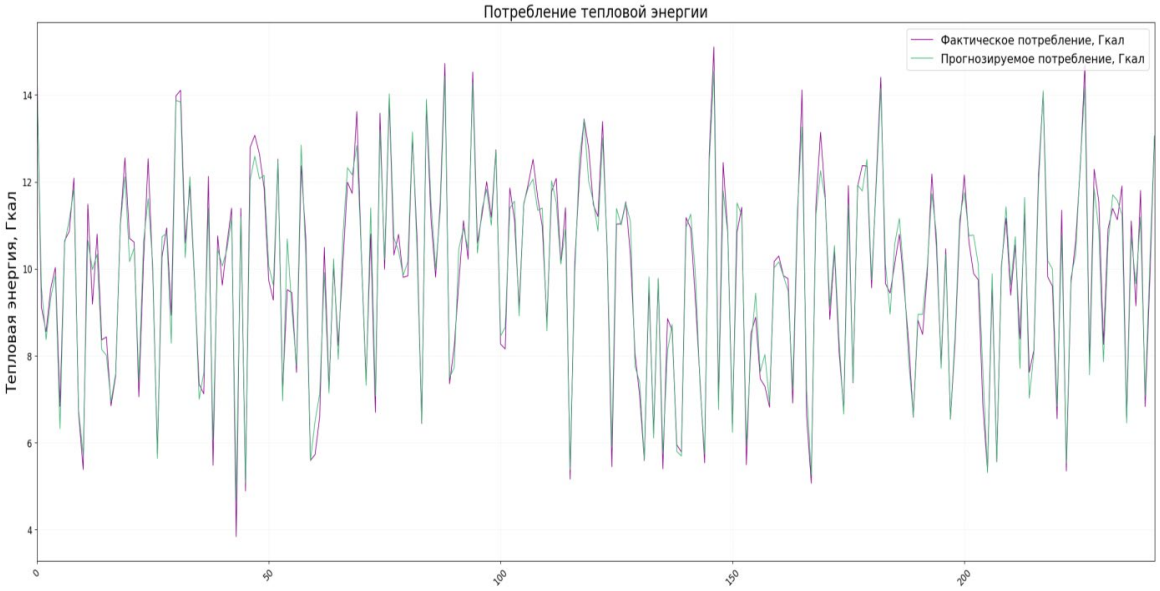


Рисунок 8 – Фактическое и прогнозируемое потребление для тренировочных данных (Random Forest)

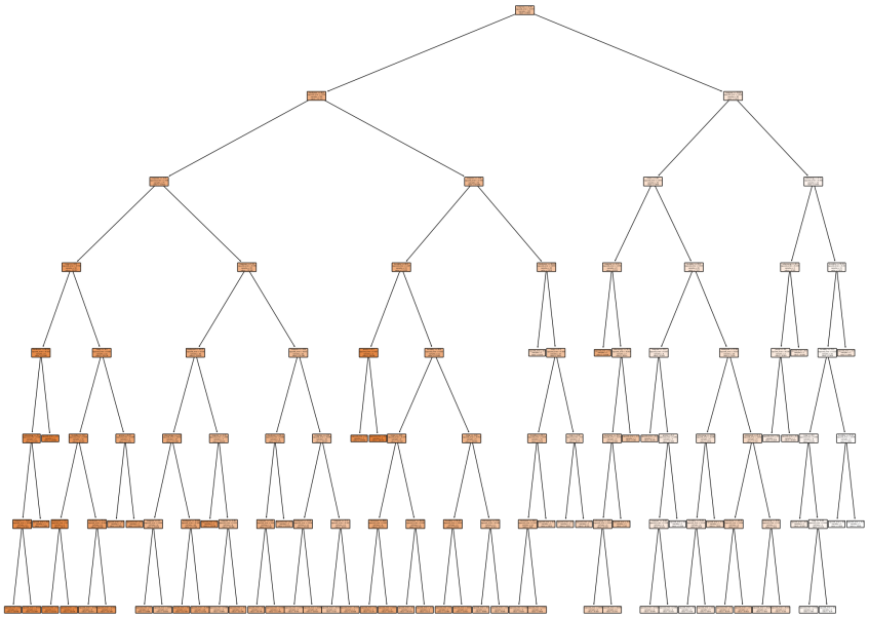


Рисунок 9 – Фрагмент дерева принятия решений для рассматриваемой модели

Анализ эффективности модели на основе алгоритма Random Forest на тестовых данных

Погодные данные и данные о потреблении тепловой энергии, используемые для тестирования, отображены на рис. 10.

Далее тестируемые данные были подставлены в созданную раннее модель, проведен расчет коэффициента детерминации и среднеквадратической ошибки (табл. 2).

График фактического и прогнозируемого потребления для тестовых данных отображен на рис. 11.

	date	temp	tempmin	tempmax	humidity	precip	windspeed	cloudcover	solarenergy	heat_cons
0	2020-02-01	1.4	-1.1	4.9	94.5	1.6	27.4	91.8	3.8	10.8047
1	2020-02-02	2.6	-1.4	6.5	94.6	2.2	36.7	86.1	3.8	8.6980
2	2020-02-03	3.7	0.2	6.6	93.7	10.5	27.4	88.5	3.6	8.0367
3	2020-02-04	0.3	-3.1	4.9	83.7	0.1	30.2	51.8	7.4	10.1735
4	2020-02-05	0.4	-1.3	1.7	96.1	30.1	32.4	93.9	1.9	11.0592
5	2020-02-06	-1.2	-4.2	0.4	93.3	22.5	41.0	91.0	3.8	11.4630
54	2020-03-26	5.4	-1.4	13.3	63.4	0.0	31.7	6.5	16.5	7.1886
55	2020-03-27	8.5	2.0	15.9	54.2	0.0	40.7	57.1	18.0	5.9058
56	2020-03-28	9.2	3.5	16.2	49.6	0.0	38.2	84.6	11.7	5.3792
57	2020-03-29	8.7	3.0	14.7	59.3	0.0	29.5	90.4	14.5	5.3247
58	2020-03-30	11.3	6.0	17.9	73.4	0.0	20.5	62.3	15.4	5.3313
59	2020-03-31	7.8	2.6	13.7	72.2	0.0	24.5	18.3	18.4	5.5425

Рисунок 10 – Данные для тестирования

Таблица 2 – Критерии эффективности модели

Параметр	Значение
Коэффициент детерминации R^2	0,89
Среднеквадратическая ошибка MSE	0,93

Рисунок 11 – Фактическое и прогнозируемое потребление для тестовых данных (Random Forest)

Несмотря не некоторое снижение эффективности для тестовых данных, разработанная модель всё равно обладает достаточно высокой эффективностью. Некоторое снижение коэффициента детерминации и повышение среднеквадратической ошибки указывают, скорее, не на недостаток модели, а на ограниченное количество данных, используемых для тренировки модели.

При увеличении тренировочных данных эффективность модели на тестовых данных будет увеличиваться.

Обучение модели для прогнозирования потребления тепловой энергии при использовании алгоритма Support Vector Machines

Фрагмент кода, с помощью которого импортируется необходимый алгоритм, вносятся некоторые параметры и тренировочные данные, а также рассчитываются коэффициент детерминации R^2 и среднеквадратическая ошибка, отображен на рис. 12.

```
[88] from sklearn.svm import SVR
     SVReg = SVR(kernel = 'rbf',C=40.0, epsilon = 0.03, gamma = 'scale') #C=1.0, epsilon = 0.1, gamma = 'scale' - default
     SVReg.fit(x_train_SVR, y_train_SVR)
```

```
SVR(C=40.0, epsilon=0.03)
```

```
Predicted_train_SVR = SVReg.predict(x_train_SVR)
Predicted_train_SVR
```

```
[17] from sklearn.metrics import r2_score
     from sklearn.metrics import mean_squared_error
```

```
[90] print(r2_score(y_train_SVR,Predicted_train_SVR))
     print(mean_squared_error(y_train_SVR,Predicted_train_SVR))
```

```
0.9019015633705991
0.548545736945683
```

Рисунок 12 – Фрагмент кода с обучением модели алгоритма *Support Vector Machines*

Значения коэффициента детерминации и среднеквадратической ошибки для тренировочных данных отображены в табл. 3. Фактической и прогнозируемое потребление тепловой энергии для тренировочных данных отображены на рис.13.

Таблица 3 – Критерии эффективности модели

Параметр	Значение
Коэффициент детерминации R^2	0,90
Среднеквадратическая ошибка MSE	0,55

.

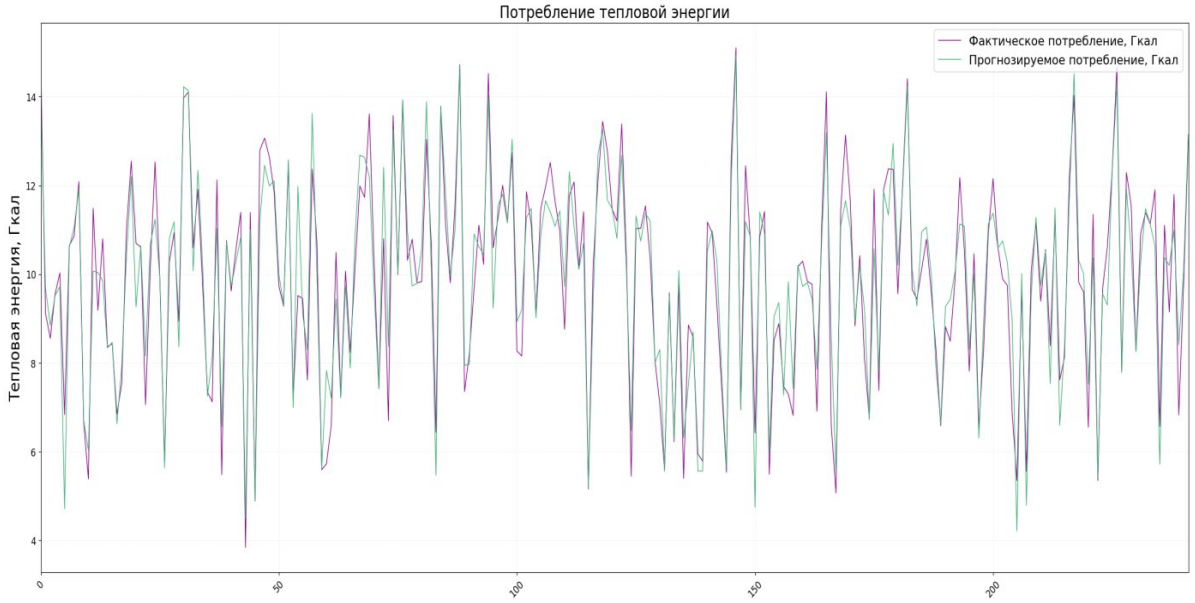


Рисунок 13 – Фактическое и прогнозируемое потребление для тренировочных данных (*Support Vector Machines*)

Исходя из результатов таблицы 3 и графика на рис. 13, разработанная модель показывает достаточно высокую эффективность на тренировочных данных. Однако величина среднеквадратической погрешности для тренировочных данных несколько выше, чем для алгоритма *Random Forest.* Далее необходимо проверить модель на тестовых данных.

Анализ эффективности модели на основе алгоритма Support Vector Machines на тестовых данных

Погодные данные и данные о потреблении тепловой энергии, которые используются для тестирования, отображены на рис. 10. Далее тестируемые данные были подставлены в созданную модель, проведен расчет коэффициента детерминации и среднеквадратической ошибки (табл. 4). График фактического и прогнозируемого потребления для тестовых данных отображен на рис. 14.

Таблица 4 – Критерии эффективности модели

Параметр	Значение
Коэффициент детерминации R^2	0,90
Среднеквадратическая ошибка MSE	0,8

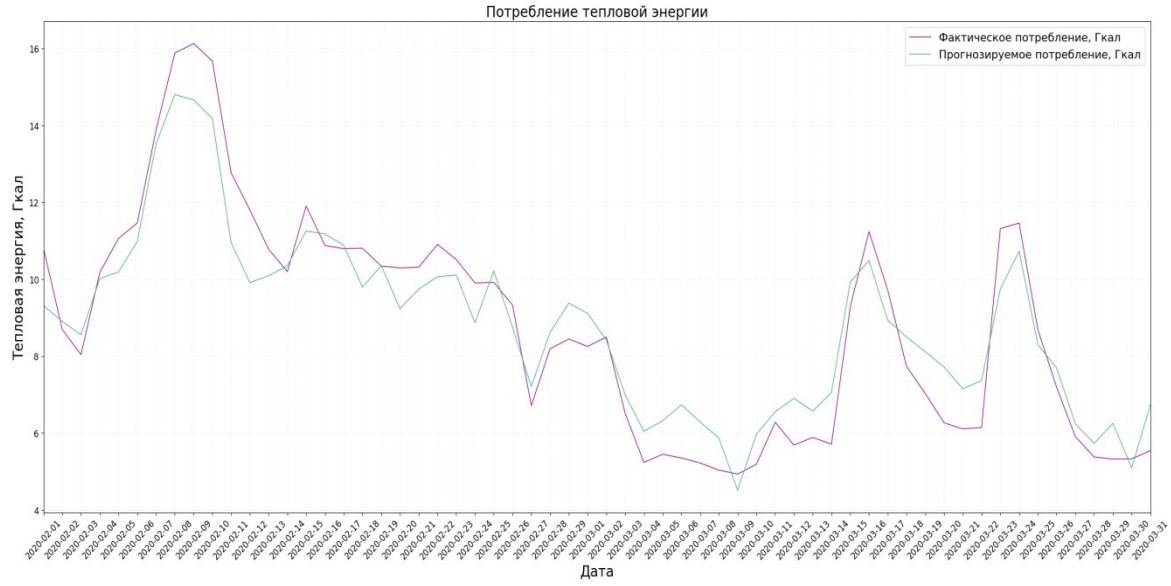


Рисунок 14 – Фактическое и прогнозируемое потребление для тестовых данных (*Support Vector Machines*)

В отличие от модели на основе алгоритма *Random Forest*, модель на основе алгоритма *Support Vector Machines* показывает несколько большую точность на тесовых данных. В любом случае, как и для модели на основе алгоритма *Random Forest,* точность предсказаний будет повышаться при увеличении количества данных для обучения модели.

Выводы

Прогнозирование потребления энергетических ресурсов имеет значительный потенциал применения в сферах энергоснабжения, энергосбережения, учета потребления энергетических ресурсов [6-9]. Современные ресурсы позволяют анализировать значительные объемы данных и создавать модели, которые с достаточно высокой точностью способны прогнозировать потребление энергетических ресурсов для отдельных объектов или комплексов [10].

В данной работе были построены модели прогнозирования потребления тепловой энергии объектом сферы здравоохранения в зависимости от погодных данных на основе алгоритмов машинного обучения. При этом набор входных данных для обучения моделей был весьма ограничен (непосредственно для обучения модели были использованы данные 246 суток).

В качестве алгоритмов были использованы *Random Forest* и *Support Vector Machines* т.к. именно эти алгоритмы более всего подходят для решения задач регрессии. Точность разработанных моделей проверялась и сопоставлялась на тренировочных и тестовых данных.

Было установлено, что обе модели показывают достаточно высокую эффективность в условиях ограниченного количество данных для обучения. При этом для алгоритма *Support Vector Machines* показывает себя несколько лучше именно на тестовых данных (значение среднеквадратической ошибки ниже, отличие между критериями эффективности для тренировочных и тестовых данных также ниже).

В любом случае, оба алгоритма имеют значительный потенциал для построения моделей прогнозирования потребления энергетических ресурсов.

Литература

1. Копейка, Д. В. Определение параметров влагопередачи и расположения плоскости возможной конденсации в системах навесных фасадов с вентилируемой воздушной прослойкой / Д. В. Копейка, С. В. Гридин // Современное промышленное и гражданское строительство. – Макеевка: ДонНАСА, 2019. – Т. 15. - №1. – С. 5–11.

2. Елисеева, И. И., Юзбашев М. М. Общая теория статистики: учебник / Под ред. И.И. Елисеевой. - 5-е изд., перераб. и доп. - М.: Финансы и статистика, 2004. — 656 с.

3. Larry Wasserman. All of statistics: a concise course in statistical inference / Larry a. Wasserman. 1959.

4. Andreas C. Mueller and Sarah Guido. Introduction to Machine Learning with Python. 2006.

5. Sebastian Raschka. Python Machine Learning. 2015.

6. Сергеев, Н. Н. Теоретические аспекты энергосбережения и повышения энергетической эффективности промышленных предприятий / Н. Н. Сергеев // Вестник АГТУ. Сер. : Экономика, 2013. - №1. - С. 29-36.

7. Колосов, М. В. Л. Н. Энерго- и ресурсосбережение в системах централизованного теплоснабжения / М. В. Колосов, Л. Н. Борисов // 11-я Межд. ИНК «Проблемы энергосбережения и экологии в промышленном и жилищно-коммунальном комплексах». - Пенза, 2010. - С. 128-130.

8. Копейка, Д. В. Повышение энергоэффективности жилых и административных зданий типовых серий / Д. В. Копейка, С.В. Гридин // Энергия – 2016: материалы XI-й межд. науч.-техн. конф. студ., асп. и мол. учёных, 5-7 апреля 2016 г., РФ – Иваново: Ивановский ГЭУ, 2016.– С. 125-131.

9. Копейка, Д. В. Определение параметров теплового комфорта в помещении при регулировании тепловой нагрузки централизованного теплоснабжения // Д. В. Копейка // Материалы 6-ой международной научно-практической конференции молодых ученых и студентов в рамках 12-ой международной конференции по проблемам горной промышленности, строительства и энергетики «Опыт прошлого – взгляд в будущее». - Тула: ТулГУ, 2016. - С. 255-258.

10. Бирюков, А. Б. Методика оперативного сбора данных для анализа энергоэффективности теплоснабжения общественных зданий / А. Б. Бирюков, А. Ю. Харитонов // Энергетические, управляющие и информационные системы: сб. докладов I-ой межд. научно-техн. конф. – Белгород: Изд-во БГТУ им. В.Г. Шухова, 2016. – С. 40-45.

Бирюков А. Б., Гридин С. В. Моделирование и прогнозирование потребления тепловой энергии с использованием погодных данных. *Работа посвящена моделированию и прогнозированию потребления тепловой энергии объектом сферы здравоохранения в зависимости от данных о погодных условиях на основе алгоритмов машинного обучения. В качестве алгоритмов были использованы Random Forest и Support Vector Machines, так как именно эти алгоритмы более всего подходят для решения задач регрессии. Точность разработанных моделей проверялась и сопоставлялась на тестовых данных. Модели, разработанные на основе указанных алгоритмов, показывают достаточно высокую эффективность в условиях ограниченного количества данных для обучения и имеют значительный потенциал для использования.*

Ключевые слова: *моделирование, тепловая энергия, алгоритм, машинное обучение.*

Biryukov A. B., Gridin S. V. Modeling and forecasting of thermal energy consumption using weather data. *The work is devoted to modeling and forecasting the consumption of thermal energy by a healthcare facility depending on weather data based on machine learning algorithms. Random Forest and Support Vector Machines were used as algorithms, since these algorithms are most suitable for solving regression problems. The accuracy of the developed models was checked and compared on test data. The models developed on the basis of these algorithms show sufficiently high efficiency in conditions of a limited amount of training data and have significant potential for use.*

Keywords: *modeling, thermal energy, algorithm, machine learning.*

Статья поступила в редакцию 12.04.2024
Рекомендована к публикации профессором Павлышом В. Н.

УДК 004.8

Проектирование расширяемой библиотеки классов компонент атрибутов из алгебры кортежей на C#

И. Д. Оверчук, О. Ю. Чередникова
Донецкий национальный технический университет, г. Донецк
e-mail: ivan.overchuk@gmail.com

Аннотация
В статье освещается процесс разработки расширяемой библиотеки классов математических структур под названием компоненты атрибутов. Описание работы сопровождается обоснованиями принятых решений в соответствии с нуждами проекта и рекомендованными паттернами проектирования. Шаблоны объектно-ориентированного программирования позволяют создать многофункциональную библиотеку классов компонент атрибутов с дальнейшей возможностью расширения. Также было освещено применение таких новшеств языка C#, как классы-примеси на основе интерфейсов с реализацией методов по умолчанию.

Общая постановка проблемы

Алгебра кортежей (АК) – это математический аппарат, разработанный Б.А. Куликом в соавторстве в А.Я. Фридманом и описанный в работах [1-6]. Этот математический аппарат относится к классу булевых алгебр и позволяет реализовать алгебраический подход к логическому анализу в системах искусственного интеллекта. В АК, в отличие от формальных систем, где основа – символьные конструкции, в качестве базового выбрано понятие многоместного отношения и предложены обобщения операций алгебры множеств для работы с отношениями, заданными в разных схемах. Основными математическими структурами алгебры кортежей (АК), над которыми производятся операции, подобные операциям над множествами, являются кортежи и системы кортежей.

C-кортеж представляет собой многоместное отношение над типом объектов. C-система – это объединение C-кортежей.

D-кортеж представляет собой объединение одноместных отношений над типом объектов. D-система – это пересечение D-кортежей.

Кортежам и системам кортежей соответствуют такие объекты, как схемы. Схема – это набор свойств, принадлежащих объектам и участвующих в отношении. Эти свойства называются атрибутами кортежей, а их значения (наборы значений) – компонентами атрибутов. И несмотря на то, что в АК кортежи являются элементарными структурами (над компонентами атрибутов операции отдельно не проводятся), на программном уровне вычисления производятся непосредственно именно над компонентами атрибутов. Кортежи представляют собой декартово произведение значений компонент атрибутов.

Компоненты атрибутов бывают пустые, полные и нефиктивные. Пустые и полные описывают соответствующие наборы значений, нефиктивные – непустые и неполные. Под полнотой здесь понимается описание компонентой всех значений из домена атрибута, называемых универсумом.

Иллюстрация математических структур алгебры кортежей приведена на рис. 1.

Постановка задачи

В данной статье освещается процесс разработки расширяемой библиотеки классов компонент атрибутов. Она является частью дипломной работы по программной реализации алгебры кортежей. Целью работы является разработка системы логического вывода в парадигме объектно-ориентированного программирования.

Разработка велась на C# 11.0. Библиотека классов покрыта модульными тестами, которые доказали её работоспособность.

Основные положения реализации библиотеки классов

Классы компонент атрибутов описывают наборы значений, которые затем потребляются классами кортежей, чтобы иметь возможность перечислять сущности со свойствами, имеющими соответствующие значения. Из этого следует, что компоненты атрибутов обязаны реализовывать интерфейс IEnumerable<T>, где T – тип хранимых данных. Родительский абстрактный класс – AttributeComponent<T>, определяющий стандартное поведение при вызове свойств компоненты и методов операций, а также абстрактный метод перечисления хранимых данных.

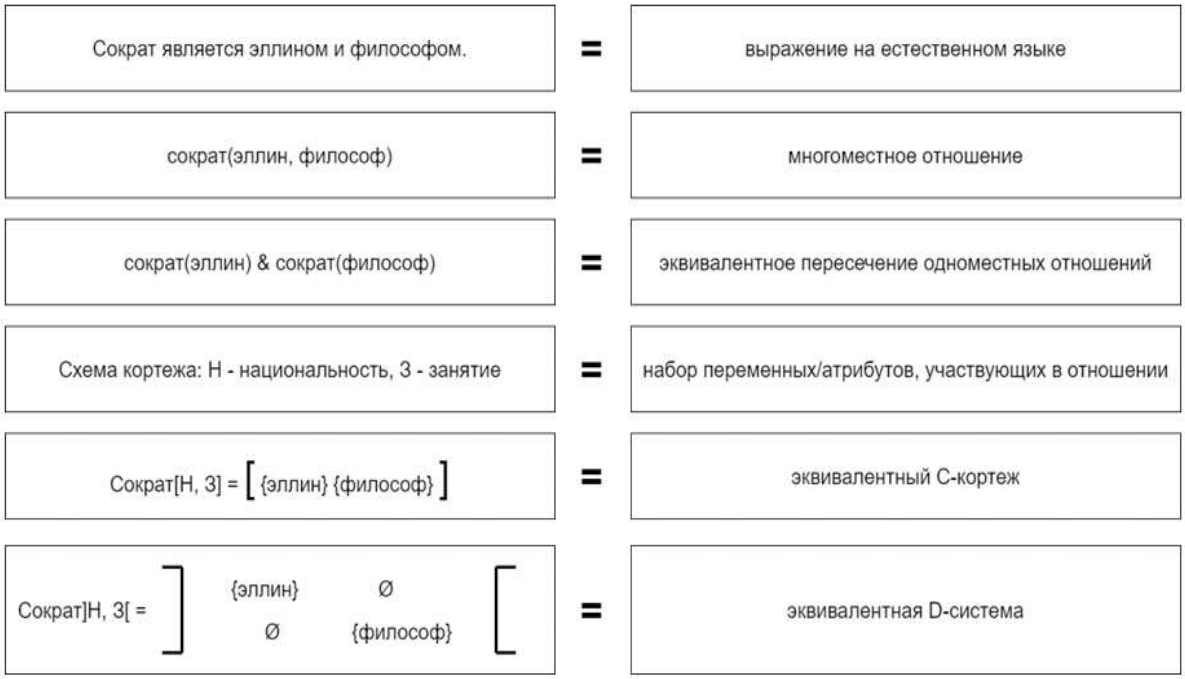


Рисунок 1 – Пример перевода выражения на естественном языке на язык АК

Каждой компоненте соответствуют следующие объекты:

Мощность компоненты: показывает её величину, если это применимо. Пассивно позволяет сравнивать типы компонент (пустая < нефиктивная < полная).

Домен атрибута: компонента должна описывать набор значений из определённого домена. Разделяется компонентами, ему принадлежащими.

Контейнер операторов: вследствие большого разнообразия видов компонент и их комбинаций в операциях набор операторов и сами операторы для каждого класса выделяются в отдельные типы. Разделяется компонентами одного класса.

Фабрика, которая произвела данную компоненту.

Каждый класс компоненты, если это необходимо, имеет статический конструктор, в котором вызывается метод регистрации объектов фабрики и контейнера операторов в качестве ресурсов, разделяемых экземплярами класса. Метод регистрации находится в отдельном классе-регистраторе (рис. 2). Такой механизм необходим для того, чтобы не держать ссылки на фабрики и контейнеры операторов в каждой компоненте. Располагать эти ресурсы в статических свойствах не удастся по причине высокой разветвлённости дерева наследования.

Создание компонент атрибутов следует паттерну «абстрактная фабрика» [7]. Каждый класс фабрики переопределяет методы, связанные с созданием нефиктивных компонент; методы создания пустых и полных компонент являются запечатанными. Каждой фабрике соответствует домен атрибута. Все компоненты, продуцируемые этой фабрикой, принадлежат этому домену. Это решение продиктовано нуждами библиотеки классов кортежей и имеет смысл по причине многофункциональности фабрик компонентов.

Реализация пустых и полных компонент атрибутов

Классы пустой (Empty) и полной (Full) компонент позволяют значительно сократить время выполнения всех операций, поскольку их результаты заведомо известны. Вместо постоянных проверок в коде компонент на предмет пустоты или полноты работа с проходящими проверку идёт как с отдельными нефиктивными компонентами

Реализация нефиктивных компонент атрибутов

Прежде всего стоит отметить, что слова «атрибуты компонентов описывают наборы значений» подразумевают различные методы описания, которые имеют свои цели, преимущества и недостатки. На данный момент библиотека классов нефиктивных (NonFictional) компонент атрибутов содержит следующие:

Итерируемые (Iterable). Такие компоненты создаются на основе перечислений IEnumerable<T>, которые предоставляют удобный инструмент для создания генерирующих последовательностей и конечных автоматов.

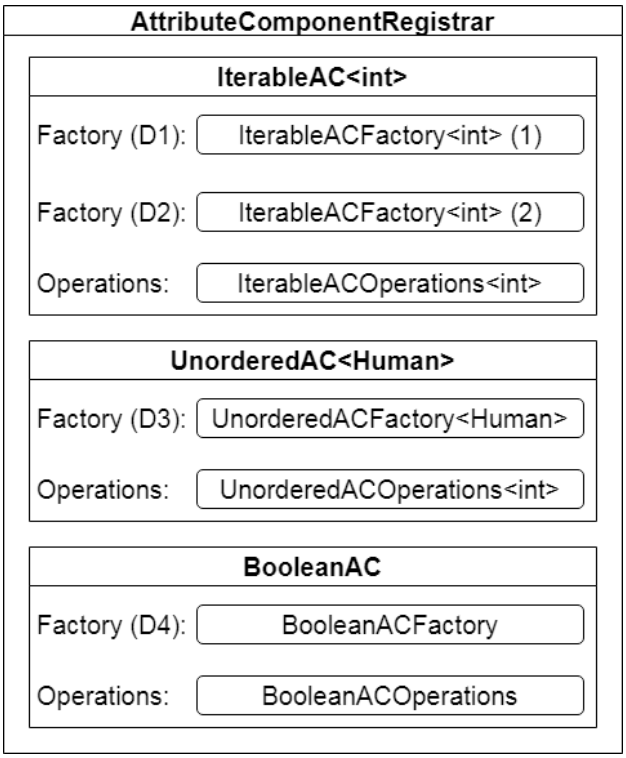


Рисунок 2 – Хранение разделяемых ресурсов в регистраторе классов компонент атрибутов

Преимуществом такого подхода является отсутствие затрат на хранение данных (хранится только код для генерации последовательности) вкупе с минимальными временными затратами на создание объектов. Минус – эффективно использоваться такая структура данных при генерации, а не извлечении из другого перечисления, может только на нессылочных типах данных (структурах), ссылочные будут нагружать сборщик мусора при больших объёмах данных вследствие создания большого числа объектов, подлежащих сборке мусора. Пример использования: генерация числовых рядов.

Неупорядоченные (Unordered). Такие компоненты создаются на основе хеш-таблиц HashSet<T>. Их преимуществом является малое время выполнения операций вида Unordered X Unordered (а также операции дополнения), в общем случае лежащее в пределах [$O(n_1)$, $O(n_1 + n_2)$]. Недостатки те же, что и у хеш-таблиц: затраты на хранение ключей и поиск данных в случае коллизий либо требование к наличию хорошей хеширующей функции. Пример использования: списки персонала организации.

Упорядоченные (Ordered). Такие компоненты создаются на основе любых структур данных, реализующих интерфейс IEnumerable<T>, и компаратора IComparer<T> для упорядочивания данных. Их преимущество – это возможность обходиться почти без затрат на хранение данных благодаря механизму доступа к непрерывной области памяти, реализованному в структурах ReadOnlySpan, ReadOnlyMemory. Для достижения этой цели были разработаны два класса: BufferingOrdered (буферизирующие) и StreamingOrdered (потоковые). Буферизирующие компоненты хранят данные в памяти в виде массива, который представляет собой непрерывную область памяти, и используются в качестве универсума домена атрибута. Потоковые компоненты оптимизируют память при помощи объекта класса ReadOnlySequence – связного списка экземпляров ReadOnlyMemory, областей непрерывной памяти. Операции сравнения вида Ordered X Ordered производятся крайне быстро вследствие упорядоченности ($O(\max(n_1, n_2))$), а прочие операции передают на выход уже отсортированные данные, т. е. сортировать данные в каждом домене нужно только раз – для универсума. Из недостатков можно выделить то, что упорядоченные компоненты эффективно работают лишь при наличии упорядоченного универсума домена и кооперации с себе подобными, а также проблемы потоковых компонент: в худшем случае связный список ReadOnlySequence будет содержать столько же элементов ReadOnlyMemory, сколько элементов представляет компонента (такую редукцию можно устранить путём замены с определённого этапа потоковой компоненты на буферизирующую). Поскольку упомянутые типы для чтения областей памяти обладают довольно скудным программным интерфейсом, был разработан специальный статический класс ReadOnlySequenceHelper для преобразования

ReadOnlySequence в отсортированную по эталону последовательность и наоборот. Пример использования: большие массивы числовых данных.

Булевые (Boolean). Пример специализированной для определённого типа компоненты. Фабрика содержит две константных переменных для экономии ресурсов. Контейнер операций оперирует непосредственно над булевыми значениями. Мощность всегда показывает единицу. Подобное решение можно экстраполировать на компоненту перечислений-флагов (enum с атрибутом типа FlagsAttribute). Плюс такой реализации – большая эффективность.

Фильтрующие (Filtering). Такие компоненты создаются на основе деревьев выражений (expression trees), представляющих предикаты (Predicate<T> или Func<T, bool>). Их назначение – предоставлять функционал SQL/LINQ-запроса WHERE по отношению к компонентам других типов, а также описывать набор данных не правилом генерации, а правилом фильтрации, при этом фильтруются значения из универсума домена. Преимущества следующие: деревья выражений разных фильтрующих компонент можно комбинировать в результате выполнения операций; деревья выражений в потенциале интеллектуально позволяют программе оценивать результат выражения заранее (например, выход за пределы числового ряда) или переводить в форму выражения некоторые последовательности, однако это сопряжено с трудностями реализации. Недостаток: не могут использоваться в качестве универсума домена, поскольку для этого не предназначены. Пример использования: фильтрация универсума целых чисел по признаку чётности.

Также имеются интерфейсы IFiniteEnumerable<T> и ICountable<T>. Первый сообщает лишь о наличии перечисления значений (отсутствует у фильтрующих компонент), второй наследуется от первого и сообщает о наличии у компоненты вычисленного количества хранимых данных (неприменимо к итерируемым компонентам). Эти интерфейсы позволяют обобщить операции на несколько классов компонент.

Реализация мощностей компонент

Класс AttributeComponentPower служит для инкапсуляции операций сравнения двух компонент по количеству хранимых данных (если применимо) и равенства мощности данной компоненты мощности пустой или полной. Первая операция может применяться в алгоритмах операций алгебры множеств, вторая используется фабриками компонент атрибутов для определения необходимости преобразования нефиктивной компоненты в пустую или полную. Соответственно, необходимо проектировать подклассы AttributeComponentPower для каждого подтипа нефиктивной компоненты. Класс AttributeComponentPower возможно реализовать по шаблону «приспособленец» [3], поскольку мощности пустых и полных компонент не требуют внешнего контекста, а для мощностей нефиктивных компонент таким контекстом выступают сами нефиктивные компоненты. Такая реализация значительно уменьшит число хранимых в памяти экземпляров AttributeComponentPower.

Реализация доменов атрибутов

Класс домена атрибута AttributeDomain<T> – это обёртка над классом AttributeComponent<T>. Он может выступать членом любых бинарных операций вместе с другой компонентой, выполнение которых он делегирует свойству Universe – универсуму домена, который представляет собой нефиктивную компоненту. Выделение этого класса необходимо для разделения «полных» нефиктивных компонент и «неполных». Также этот класс позволяет добавить функцию именования доменов (что полезно, например, для хеширования доменов или их идентификации).

Реализация фабрик нефиктивных компонент атрибутов

Классы фабрик наследуются от родительского класса AttributeComponentFactory<T>. Этот класс определяет методы создания пустых и полных компонент, а также шаблонный метод создания нефиктивных: если после создания компоненты её мощность показывает пустоту или полноту, метод возвращает пустую или полную компоненту соответственно.

Фабрики потребляют экземпляры подкласса фабричных аргументов AttributeComponentFactoryArgs, которые содержат необходимые данные (перечисление значений, выражение предиката, мощность компоненты и т. д.). Каждому неабстрактному и использующемуся в коде классу нефиктивной компоненты соответствует такой подкласс (методы создания пустых и полных компонент требуют экземпляр AttributeComponentFactoryArgs, поэтому возможна передача фабричных аргументов для создания нефиктивных компонент). В языке C# отсутствует поддержка множественного наследования, однако его функционал необходим для возможности расширения описываемой библиотеки классов. Поскольку в первую очередь расширяется список классов нефиктивных компонент и требуется поддержка

этих новых классов в других модулях, нельзя статично закодировать их создание и использование, иначе потребуется постоянная рекомпиляция кода, что, в сущности, и противоречит принципу расширяемости. Однако C# предлагает иное, элегантное решение: классы-примеси на основе интерфейсов с реализацией методов по умолчанию. Благодаря этому имеется возможность проектировать классы-примеси фабрик, которые могут создавать нефиктивные компоненты различных видов: упорядоченные, итерируемые, фильтрующие. Данный момент очень важен в библиотеке классов кортежей: при настройке схемы кортежей необходимо указывать фабрики для каждого атрибута.

Подклассы фабрик реализуют интерфейс INonFictionalAttributeComponentFactory<T, TFactoryArgs>. Этот интерфейс определяет метод создания нефиктивной компоненты с определённым подклассом фабричных аргументов. Интерфейсы, наследующие INonFictionalAttributeComponentFactory<T, TFactoryArgs>, добавляют реализацию по умолчанию для этого метода.

Создание кортежей основывается на передаче их фабрикам экземпляров подклассов AttributeComponentFactoryArgs, поэтому следует обеспечивать фабрики компонент всеми нужными возможностями для работы с фабричными аргументами. Однако, так как кортежи принимают экземпляры базового класса AttributeComponentFactoryArgs, кортежи не могут знать, какой интерфейс фабрики соответствует каждому фабричному аргументу, поэтому передача аргумента в фабрику делегируется самому аргументу. Данное поведение также реализуется при помощи примесей.

Реализация контейнеров операторов компонент атрибутов

Компоненты атрибутов поддерживают такие операции алгебры множеств, как отрицание, объединение, пересечение, разность и симметрическую разность, проверку включения одной компоненты в другую, равенства друг другу и включения или равенства. Почти все из этих операций являются бинарными и допускают многие сочетания типов компонент атрибутов, и все эти операции имеют различную реализацию для разных сочетаний. По причине большого числа сочетаний, которое может со временем увеличиваться при расширении библиотеки классов, следует использовать паттерн «Состояние» и вынести все бинарные операции в отдельные классы операторов, которые реализуют паттерн ООП «Посетитель» с дополнительной диспетчеризацией для второго аргумента [7, 8].

Можно выделить две группы бинарных операторов: с немедленным, заведомо известным результатом (в случае пустых и полных компонент) и с вычисляемым, использующим фабрику (для нефиктивных). У первой группы суперкласс называется InstantBinaryOperator, у второй – FactoryBinaryOperator. Их суть одинаковая, отличается лишь сигнатура вызываемых методов: у первой группы в сигнауру входят только два операнда, у второй добавляется фабрика, которая производит результат операции.

Чтобы операции хранились в одном месте, проектируются классы контейнеров операторов. Они также делятся на InstantOperatorContainer и FactoryOperatorContainer, каждому типу контейнеров соответствует тип операторов алгебры множеств. Операторы сравнения реализуются на базе InstantBinaryOperator, поскольку их результатом выступает булево значение. Для каждого подкласса компоненты атрибута создаётся свой подкласс контейнера операторов. FactoryOperatorContainer также содержит фабрику связанной компоненты, поскольку есть возможность избежать её постоянного поиска через регистратор типов компонент.

Есть два уровня операций: «полная/пустая/нефиктивная X полная/пустая/нефиктивная» и «конкретная нефиктивная X конкретная нефиктивная». Методы операций первого уровня реализованы в родительских классах операций вида CrossTypeOperation, методы операций второго уровня реализуются в подклассах суперклассов операторов. Допустимо дополнять их примесями:

Iterable используют интерфейсы вида IFiniteEnumerableBinaryOperation, которые реализуют операции вида IFiniteEnumerable X IFiniteEnumerable и IFiniteEnumerable X Filtering.

Unordered реализуют операции вида Unordered X IFiniteEnumerable (при этом эффективность операций Unordered X Unordered никуда не девается по причине внутренней реализации методов операций над множествами у HashSet<T>) и Unordered X Filtering.

Ordered реализуют операции вида Ordered X Ordered, Ordered X ICountable, а также используют интерфейсы IFiniteEnumerableBinaryOperation.

Boolean реализуют операции вида Boolean X Boolean.

Filtering реализуют операции вида Filtering X Filtering.

Реализация регистратора типов компонент атрибутов

Регистратор типов – это древовидная структура, соответствующая иерархии

регистрируемых типов. Каждому типу компоненты атрибута AttributeComponent<T> соответствует контейнер операторов AttributeComponentOperatorContainer<T> и множество активных доменов AttributeDomain<T>, каждый домен связан с фабрикой AttributeComponentFactory<T>. Регистратор предназначен для централизованного хранения вышеперечисленных объектов, поскольку децентрализованное будет обходиться слишком дорого по памяти. Поиск объектов, соответствующих экземплярам компонент атрибутов, производится следующим образом:

- контейнер операторов ищется лишь исходя из типа компоненты атрибута;
- фабрика ищется по типу и по домену компоненты атрибута.

Пример хранилища регистратора типов компонент атрибутов приведён на рис.3.

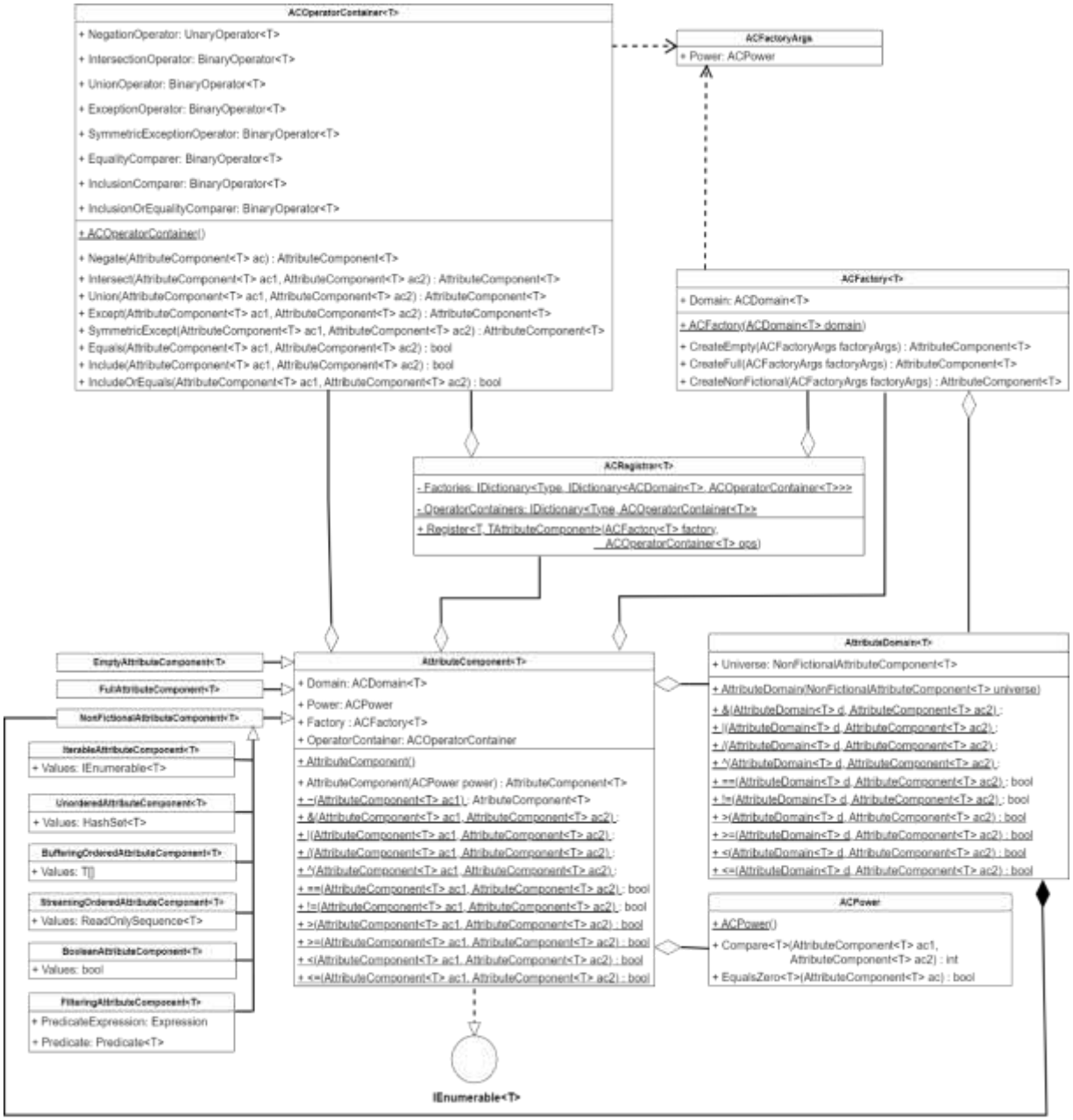


Рисунок 3 – Упрощённая UML-диаграмма классов для компонент атрибутов

При регистрации типов возможно не указывать один или оба объекта для создания (они инициализируются лениво, по мере надобности), в таком случае экземплярам этого типа будут соответствовать:

- контейнер операторов ближайшего типа-предка;
- фабрика ближайшего типа-предка с таким же доменом, что и у экземпляра.

Заключение

Паттерны объектно-ориентированного программирования позволяют создать многофункциональную библиотеку классов компонент атрибутов с дальнейшей возможностью расширения. Начальные этапы работы описаны в [9]. В данной статье освещено применение таких новшеств языка C#, как классы-примеси на основе интерфейсов с реализацией методов по умолчанию.

Литература

1. Кулик, Б. А. Логика и математика: просто о сложных методах логического анализа / Б.А. Кулик; под общ. ред. А. Я. Фридмана. – СПб. : Политехника, 2020. – 141 с. : ил.

2. Кулик, Б. А. Алгебраический подход к интеллектуальной обработке данных и знаний / Б. А. Кулик, А. А. Зуенко, А. Я. Фридман. – СПб.: Изд-во Политехн. ун-та, 2010. – 235 с.

3. Кулик, Б. А. Расширение возможностей логического анализа за счет уточнения интерпретации исчисления предикатов / Информатика и кибернетика. – Донецк: ДонНТУ, 2022. – № 3(29). - С. 5-14.

4. Kulik, B., Fridman, A. Complicated Methods of Logical Analysis Based on Simple Mathematics. – Newcastle upon Tyne: Cambridge Scholars Publishing, 2022. – 195 p.

5. Кулик, Б. А. Исследование противоречий в естественных рассуждениях на примерах метафор и пресуппозиций // Труды Семнадцатой Национальной конференции по искусственному интеллекту с международным участием. КИИ-2019 (21–25 октября 2019 г., Ульяновск, Россия). – Ульяновск: УлГТУ, 2019. Т. 2. – С. 192-200.

6. Кулик, Б. А. Вывод следствий с предварительно заданными свойствами // Системный анализ в проектировании и управлении. Материалы XXV Международной научной и учебно-практической конференции, 13-14 октября 2021 г. – СПб.: ПОЛИТЕХ-ПРЕСС, 2021. Часть 2. – С. 89-97.

7. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р Хелм, Р. Джонсон, Дж. Влиссидес. — СПб.: Питер, 2015. — 368 с.: ил.

8. Мартин, Р. Принципы, паттерны и методики гибкой разработки на языке C# / Р. Мартин, М. Мартин. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 768 с., ил.

9. Оверчук, И. Д. Программная реализация алгебры кортежей с применением методик ORM-технологий / И. Д. Оверчук, О. Ю. Чередникова // Информатика, управляющие системы, математическое и компьютерное моделирование» (ИУСМКМ-2023): сборник трудов XIV международной научно-технической конференции. – Донецк: ДонНТУ, 2023. С. 139-143.

Оверчук И. Д., Чередникова О. Ю. Проектирование расширяемой библиотеки классов компонент атрибутов из алгебры кортежей на C#. *В статье освещается процесс разработки расширяемой библиотеки классов математических структур под названием компоненты атрибутов. Описание работы сопровождается обоснованиями принятых решений в соответствии с нуждами проекта и рекомендованными паттернами проектирования. Шаблоны объектно-ориентированного программирования позволяют создать многофункциональную библиотеку классов компонент атрибутов с дальнейшей возможностью расширения. Также было освещено применение таких новшеств языка C#, как классы-примеси на основе интерфейсов с реализацией методов по умолчанию.*

Ключевые слова: *алгебра кортежей, кортеж, атрибут, домен, логические вычисления, C#, ООП, паттерны проектирования.*

Overchuk I. D., Cherednikova O. Ju. Designing an extensible class library of attribute components from the tuple algebra in C#. *The paper highlights the process of developing an extensible class library of mathematical structures called attribute components. The description of the work is accompanied by justifications of the technological decisions taken in accordance with the needs of the project and recommended design patterns. Object-oriented programming templates allow you to create a multifunctional library of attribute component classes with further extensibility. The application of such innovations of the C# language as impurity classes based on interfaces with the implementation of default methods was also highlighted.*

Keywords: *tuple algebra, tuple, attribute, domain, logical computing, C#, OOP, design patterns.*

Статья поступила в редакцию 12.04.2024
Рекомендована к публикации профессором Павлышом В. Н.

УДК 004. 4'2

Применение информационных технологий в сфере настольных ролевых игр

С. А. Айдин, А. В. Боднар
Донецкий национальный технический университет, г. Донецк
e-mail: aidin.serg@gmail.com, linabykova13@ya.ru

Аннотация
В статье рассматриваются основные понятия настольных ролевых игр, проводится анализ наиболее распространённых настольных ролевых систем и оценивается рациональность их автоматизации. Также анализируются существующие наборы инструментов и формируются требования к разрабатываемому ПО. Направлением дальнейших исследований является улучшение качества разрабатываемых систем в сфере настольных ролевых игр, что позволит упростить их проведение и повысить заинтересованность целевой аудитории настольных игр на таком жанре.

Введение

Настольные ролевые игры (НРИ) – вид настольных ролевых игр, в котором один из участников принимает на себя роль "Мастера", а остальные игроки создают себе вымышленных персонажей. Мастер (Game Master, GM, Dungeon Master, DM) – куратор настольной ролевой игры, в его обязанности входит знание базовых правил настольное ролевой системы, создание истории, подготовка сценария игры и обрабатывание запросов игроков. Основными запросами игроков в большинстве ролевых систем являются:

- вопросы, связанные с окружением игровых персонажей;
- вопросы, связанные с возможными знаниями игровых персонажей;
- попытки игрового персонажа совершить определенное действие.

Ролевая система – свод правил и рекомендаций для Мастера о том, как стоит вести игровой процесс и, что является основным отличием каждой системы, как обрабатывать запросы игроков.

Постановка проблемы

Начиная с 2022 года статистика продаж настольных игр начала понижаться [1]. Эксперты выделяют две основные причины таких изменений:

- послабление карантинных ограничений, вызванных пандемией COVID-19, из-за чего многие вернулись к своим прежним видам досуга, что были до карантинных ограничений;
- прекращение поставок настольных игр известными компаниями, в том числе связанными с настольными ролевыми играми.

Можно сказать, что настольные игры не получили своей популярности из-за того, что не смогли достаточно заинтересовать аудиторию или являлись достаточно сложными, а ограничение продукции на рынке оставило в качестве целевой аудитории только энтузиастов, изначально заинтересованных в настольных ролевых играх как жанре.

Наиболее оптимальным решением является создание инструментариев, позволяющих упростить работу мастеров, путем автоматизации основных процессов в настольных ролевых играх.

Анализ исследований и публикаций

В своей работе Д. А. Челышева проводит оценку популярности настольных ролевых игр, а также рассматривает проблему генерации локаций с помощью средств информационных технологий [2]. М. Ю. Павлов проводит описание модели взаимодействия на основе наиболее популярных настольных ролевых систем [3]. М. С. Власенко, Д. К. Фокин, В. С. Якимов и В. С. Андрианов в своей работе проводят анализ целевой аудитории настольных ролевых игр, а также выделяют проблему недостаточности инструментов для проведения данного рода игр [4]. Д. К. Беседин проводит анализ существующих веб-справочников по настольной ролевой системе GURPS, а также проводит анализ потребностей целевой аудитории системы, для которой будет разрабатываться веб-сайт [5].

Более углубленный анализ вышеописанных статей позволяет сделать вывод, что вопрос применения информационных технологий в сфере настольных ролевых игр не был решён в полной мере, что обусловило актуальность данного исследования.

Цель исследования

Целью является проектирование ПО, представляющего собой набор инструментов для

автоматизации процессов проведения настольной ролевой игры. Необходимо провести исследование существующих настольных ролевых систем и выделить аспекты проведения игр согласно данным системам. Затем выбрать из исследованных игровых систем такую, чтобы ее автоматизация была рациональна в виду её комплексности и применимости рядовыми пользователями. В заключении, требуется составить список функциональных и нефункциональных требований к разрабатываемому ПО инструментария.

Основные результаты исследования

Dungeons & Dragons (D&D, DnD; Подземелья и драконы) — настольная ролевая система в фэнтези сеттинге, разработанная Гэри Гайгэксом и Дэйвом Арнесоном. Впервые была издана в 1974 году компанией «Tactical Studies Rules, Inc.» (TSR), а с 1997 года издаётся компанией «Wizards of the Coast» (WotC). По настоящее время полноценно существуют 5 редакций. Пятая редакция D&D собрала вокруг себя наибольшую целевую аудиторию в виду своей простоты и актуального сеттинга – средневековое фэнтези.

В системе D&D главными аспектами персонажа являются его атрибуты: "Сила", "Ловкость", "Телосложение", "Интеллект", "Мудрость" и "Харизма". Данные характеристики модифицируются благодаря преимуществам выбранной расы или класса. Характеристики персонажа напрямую влияют на успешность выполняемых его действий, а также на эффективность персонажа в определенной сфере деятельности [6].

К примеру, персонаж с высоким показателем атрибута "Сила" может поднимать тяжелые объекты, переносить больший вес, а также наносить более тяжелые удары, в то время как персонаж с высоким "Интеллект" может лучше знать историю, проводить анализ и применять магию.

Вторым важным аспектом данной системы являются классы. Класс - архетип персонажа, который определяет его умения и способности, а также даёт ему преимущества для конкретного стиля игры ("Воин" хорошо сражается и является универсальным, в то время как "Бард" специализируется на применении заклинаний и поддержке других персонажей).

В игре используются кубики d4, d6, d8, d12, d20, где число - количество граней. Если ведущий считает, что выполняемое персонажем игрока действие может быть провалено, то он может попросить у игрока сделать проверку умения, которая включает в себя бонус от характеристики и другие модификаторы за сложность, назначаемые ведущим. Если сумма модификаторов и выпавшего значения на кубике превышает необходимый порог, то персонаж преуспевает при выполнении действия, в противном случае ему не удаётся выполнить запланированное, или же результат не соответствует ожиданиям игрока.

Первым наиболее успешным инструментарием для D&D является сайт dns.su (рисунок 1, таблица 1). На данном сайте находится обширная база знаний по ролевой системе: описание классов, рас персонажей, заклинаний и магических предметов, простого снаряжения и сокровищ.

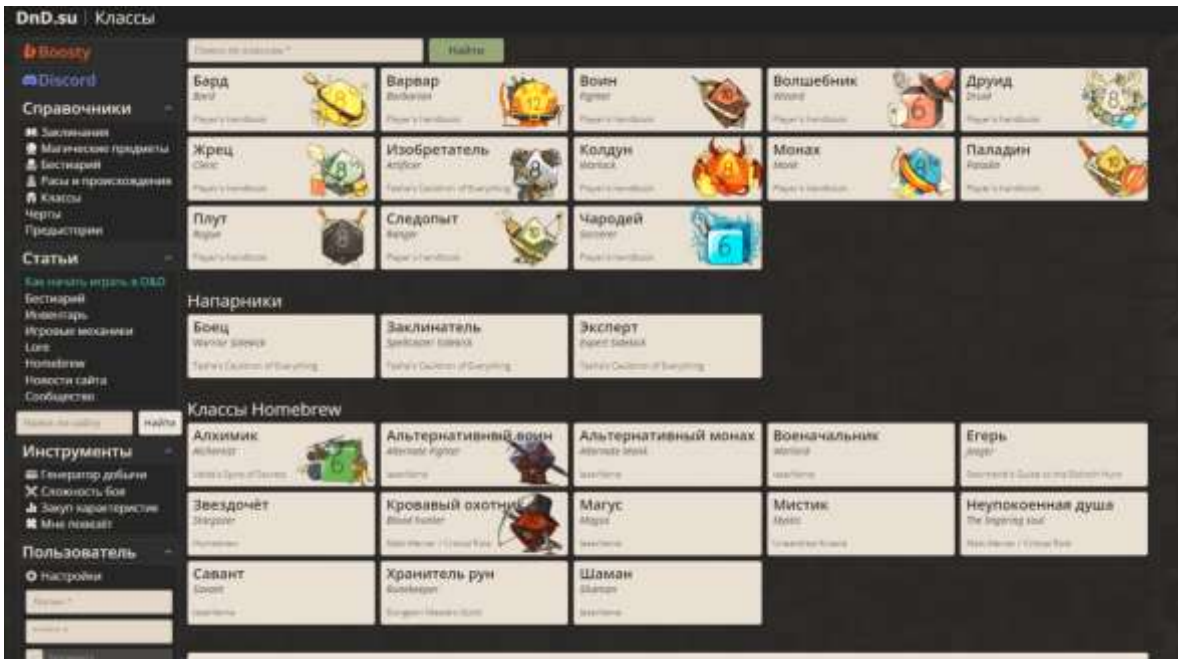


Рисунок 1 – dnd.su

Таблица 1 – Преимущества и недостатки веб-сайта dnd.su

Преимущества	Недостатки
Локализован на русский язык	Требуется подключение к интернету
Понятный и привлекательный интерфейс	
Удобство использования	
Большая база знаний	

Помимо вышеописанного, сайт представляет такие инструменты: генератор добычи, калькулятор сложности боя, подбор характеристик персонажа и "мне повезет", который откроет случайную статью из любого раздела.

Вторым инструментарием по системе D&D является приложение longstoryshort.app (рисунок 2, таблица 2), которое можно загрузить на мобильные устройства или использовать прямо из сайта. Как и предыдущий набор инструментов, данное приложение содержит в себе большое количество информации по системе, дополняет список описаниями различных правил, а также генератором листа персонажа, где игрок может вручную настроить характеристики, написать особенности и предысторию персонажу, а затем сохранить этот лист для печати или для игры онлайн.

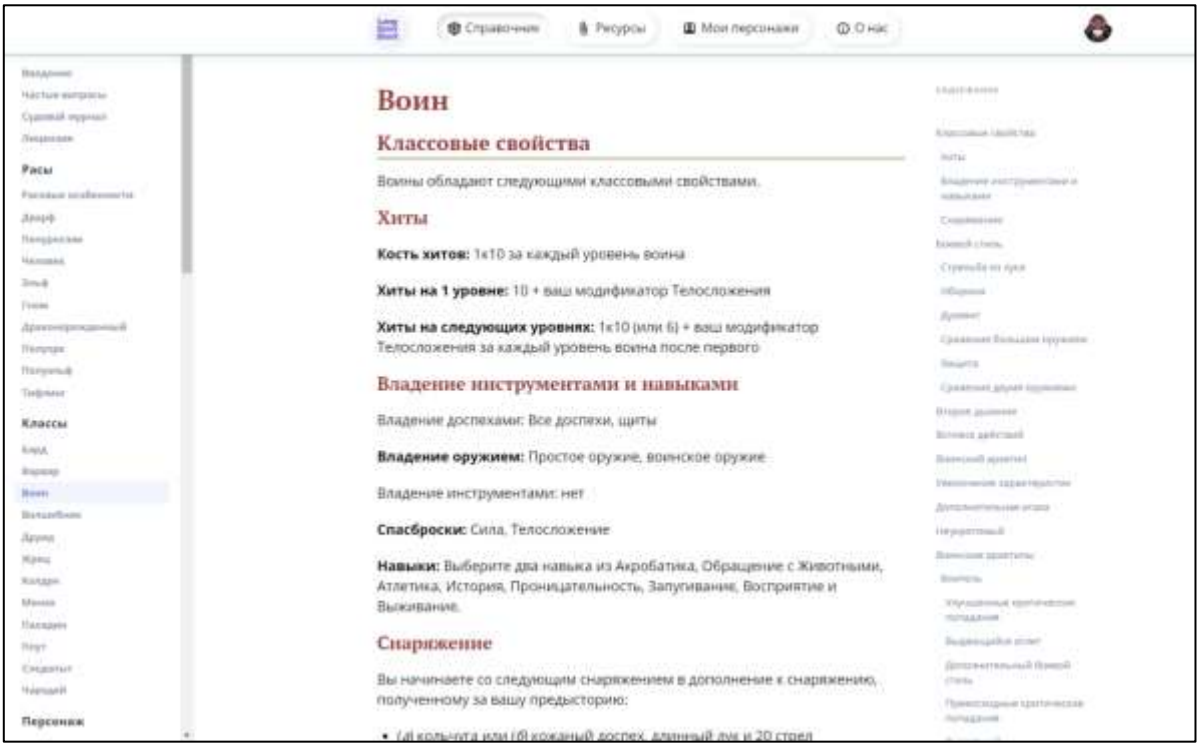


Рисунок 2 – longstoryshort.app

Таблица 2 – Преимущества и недостатки longstoryshort.app

Преимущества	Недостатки
Локализован на русский язык	Требуется подключение к интернету
Импорт и экспорт данных с устройства	Неудобная структура разделов
Большая база знаний	Непривлекательный интерфейс

Последним рассмотренным инструментом по системе D&D является десктопное приложение Aurora (рисунок 3, таблица 3) для создания и управления листом персонажа [XX]. Данное ПО распространяется на бесплатной основе и содержит в себе всю необходимую информация по основным и дополнительным книгам правил, связанную с созданием персонажа. Главным недостатком данного приложения можно назвать отсутствие локализации, что делает его менее привлекательным для русскоязычной аудитории (особенно учитывая большой объём текстовой информации в приложении).

Рисунок 3 – Aurora

Основываясь на информации выше, можно сделать вывод, что настольная ролевая система D&D не нуждается в автоматизации в виду большого количества существующих наборов инструментов и в виду своей относительной простоты. Данная ролевая система имеет обширную целевую аудиторию, для которой нет необходимости в новых инструментах.

Pathfinder Roleplaying Game (PF, PFRPG) — настольная ролевая игра в сеттинге фэнтези, разработанная Джейсоном Балманом, Джеймсом Джейкобсом, Шоном Рейнольдсом и Уэсли Шнейдер, а консультантом выступил Монте Кук. Впервые была издана в 2009 году компанией «Paizo Publishing».

Система Pathfinder (рисунок 4, таблица 4) по своей характеристике напоминает третью редакцию вышеописанной системы D&D, поскольку является её продолжителем. Основными атрибутами персонажей всё ещё являются: "Сила", "Ловкость", "Выносливость", "Интеллект", "Мудрость" и "Харизма"

Система классов в данной ролевой системе также присутствует, но теперь, кроме уникальных преимуществ, она предоставляет персонажу набор "классовых навыков", на которые он может по ходу развития добавлять "ранги", тем самым развивая свои навыки [7].

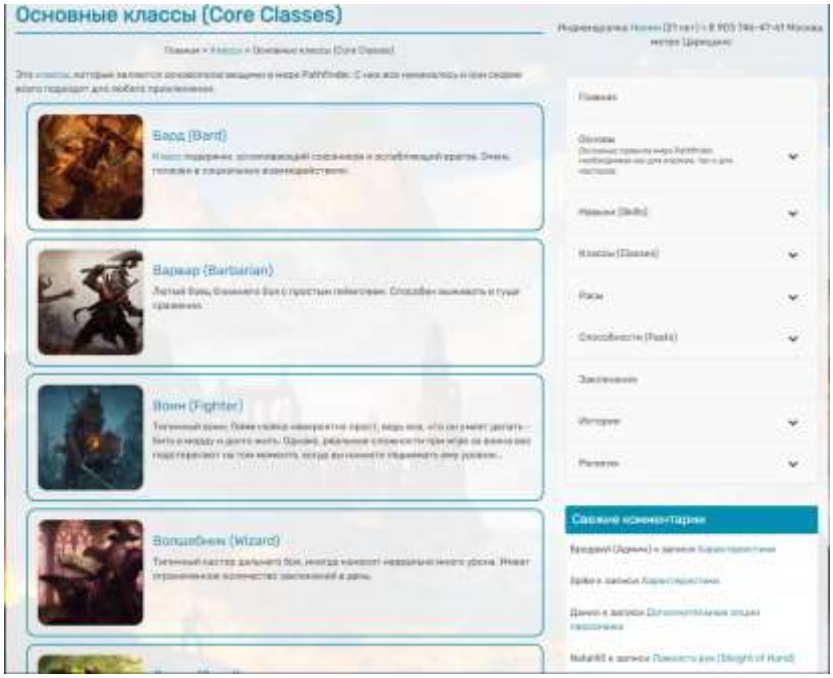


Рисунок 4 – pathfinder-wiki.ru

Таблица 4 – Преимущества и недостатки pathfinder-wiki.ru

Преимущества	Недостатки
Локализован на русский язык	Требуется подключение к интернету
Интуитивно понятный интерфейс	Неполнота информации
Удобство пользования	Долгое время отсутствует поддержка веб-сайта

Настольная ролевая система Pathfinder содержит в себе больше различных модификаторов, влияющих на броски кубиков в определенных игровых ситуациях, что делает эту систему, в некоторой мере, усложнённой по сравнению с предшествующей системой.

Первым инструментом можно выделить сайт-архив pathfinder-wiki.ru. Данный сайт содержит в себе все базовые знания по ролевой системе, однако находится на стадии разработки и долгое время не получает обновления, что делает его полезным для начального обучения, однако в дальнейшем придётся искать другой источник информации.

Основываясь на результатах исследования настольной ролевой системы Pathfinder, можно сделать вывод о нецелесообразности автоматизации данной системы. В виду своей идентичности с третьей редакцией Dungeons and Dragons, являющейся на данный момент устаревшей. К тому же, данная ролевая система уже имеют достаточную базу знаний и наборы инструментов для автоматизации большей части подготовки и игрового процесса как мастером, так и игрокам.

Call of Chtulhu (CoC; Зов Ктулху, рисунок 5, таблица 5) — настольная ролевая игра в жанре ужасы. Разрабатывается компанией Chaosium с 1981 года и на текущий момент имеет 7 редакций игровых правил. Сеттинг настольной ролевой игры основывается на условных "Мифах Ктулху" – мифопее, вымышленной литературная вселенная, берущей начало в произведениях американского писателя ужасов Говарда Филлипса Лавкрафта и более детально разработанной его последователями [8].

Системы Call of Chtulhu основана на другой ролевой системе, выпущенной от того же издателя - Basic Role Playing (BRP), - при этом была в некотором роде изменена.

Персонажи в CoC не имеют классов, а их способности зависят от развитых умений, выражаемых в процентном шансе на успех.

В качестве первичных характеристик персонажей система использует атрибуты из BRP: "Сила", "Выносливость", "Размер", "Интеллект", "Ловкость", "Сила воли", "Внешность" [9]. Кроме вышеуказанных, система Зов Ктулху добавляет новый атрибут: "Мудрость", расширяет количество вторичных атрибутов, а также вводит механику Рассудка и его потери.

Единственным комплексным набором инструментов по системе Зова Ктулху является сайт callofcthulhu.ru, представляющий собой сайт-хранилище, в котором выкладываются основные переведенные материалы, а также дополнительные ссылки на статьи в других веб-сайтах (преимущественно из одноименной группы на ресурсе vk.com). Кроме того, на сайте хранится генератор персонажа, заполняемые листы, а также большое количество переведённых и авторских сценариев для проведения игр по данной ролевой системе.

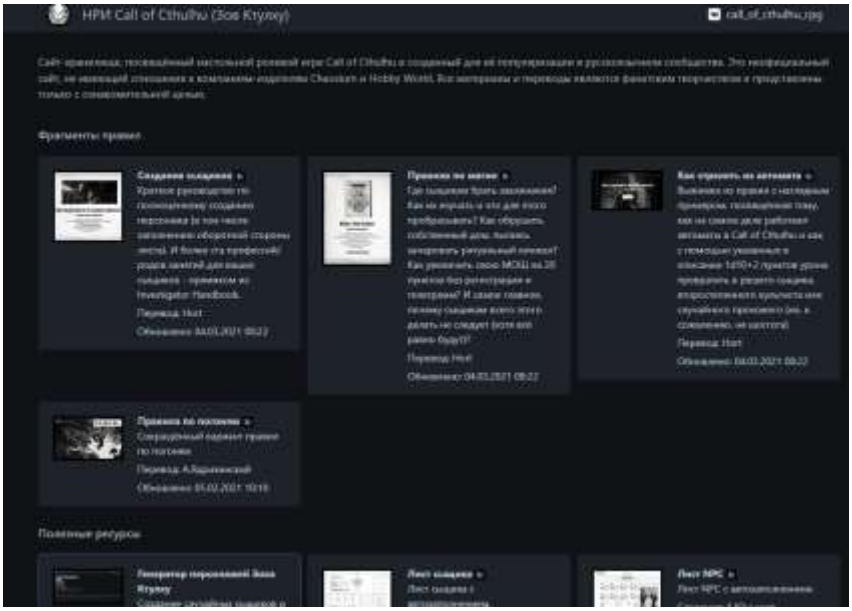


Рисунок 5 – callofcthulhu.ru

Таблица 5 – Преимущества и недостатки callofcthulhu.ru

Преимущества	Недостатки
Локализован на русский язык	Требуется подключение к интернету
Большой набор готовых сценариев для проведения игры	Малопривлекательный интерфейс
Понятный интерфейс	Плохое оформление разделов, трудность поиска необходимой информации.

В качестве итога исследования системы Call of Chtulhu можно сказать, что автоматизация данной настольной ролевой игры не рационально. Данная система ориентируется на жанре ужасов с сеттингом, напрямую связанным с вымышленной мифологией за авторством Г. Ф, Лавкрафта. Из-за этого охват целевой аудитории Зова Ктулху относительно мал, и несмотря на усложнённость системы, её автоматизация не сильно повлияет на распространённость системы среди базы любителей настольных игр.

Generic Universal Roleplaying System (GURPS; ГУРПС) — настольная ролевая игра без жанра. Была впервые издана в 1986 году компанией Steve Jackson Games и в настоящее время имеет 4 редакции, и 122 статьи в выпускаемом журнале Pyramid. Система данной настольной ролевой игры является универсальной и включает в себя наибольшее количество правил для проведения игровых сессий любого жанра и сеттинга [10].

В настольной ролевой системе GURPS персонаж характеризуется такими аспектами: основные атрибуты ("Сила", "Ловкость", "Интеллект" и "Здоровье"), вторичные атрибуты ("Единицы Жизни", "Единицы Усталости", "Восприятие", "Сила Воли", "Скорость", "Подвижность"), преимуществами, недостатками, умениями и чертами. GURPS придерживается бесклассовой системы, где все особенности персонажа покупаются за очки. Основные атрибуты влияют на вторичные, однако и те и другие можно дополнительно улучшить за очки. Преимуществами называются особые способности, которые дают непосредственный бонус персонажу (социальный статус, физические или ментальные способности). Недостатки же, выполняют обратную роль, и дают персонажу очки развития в обмен на трудности во время игрового процесса. Умения, как и в вышеописанных ролевых системах, применяются при бросках проверок, когда у персонажа есть шанс провалить собственное действие. Проверки характеристик осуществляются таким образом: игрок бросает 3 кубика d6 (6 граней) и суммирует их результат. Если полученный результат меньше или равен сложности проверки (умения или характеристики) - то персонаж преуспел в действии, а эффективность его успеха измеряется в "ступенях успеха" - разнице между порогом сложности проверки и результатом брошенных кубиков. В случае, если результат превышает порог сложности, то действие считается проваленным, а степень провала определяется "степенями провала", равному разнице выброшенного на кубиках результата от порога сложности.

Первым наиболее известным инструментом можно назвать десктопное приложение GURPS Character Sheet (GCS, рисунок 6, таблица 6).

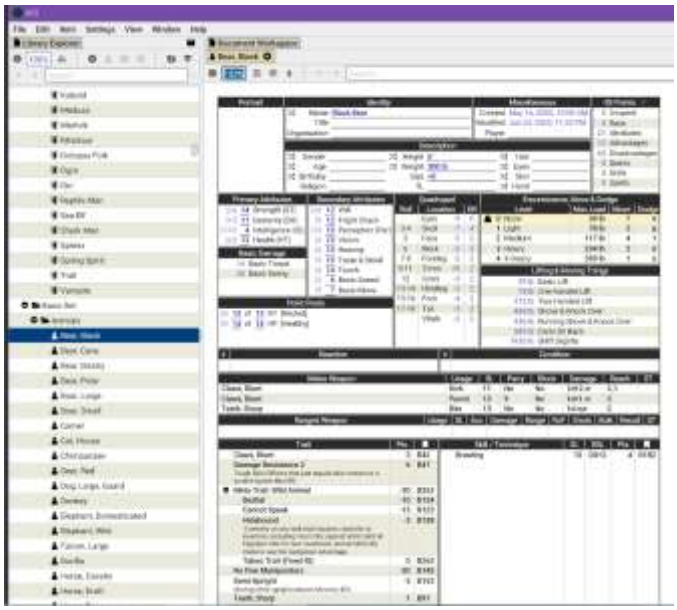


Рисунок 6 – GURPS Character Sheet

Таблица 6 – Преимущества и недостатки GURPS Character Sheet

Преимущества	Недостатки
Не требуется подключение к интернету	Отсутствие русской локализации
Большая база знаний	Отсутствие описания атрибутов
Автоматический расчёт характеристик	Малопривлекательный интерфейс
Регулярные обновления	
Экспорт/импорт данных с устройства в систему	

Данный продукт распространяется на бесплатной основе и обновляется по настоящее время. Приложение позволяет создавать персонажей, шаблоны и т.д. База информации продукта состоит из большого количества дополнительной литературы по НРИ GURPS, однако главным недостатком для пользователя можно назвать отсутствие описания преимуществ, недостатков, умений и черт – требуется обращение к книге правил, указанной в источниках атрибута.

Наиболее популярным среди русскоязычной целевой аудитории настольной ролевой системы GURPS инструментов для создания персонажа и его управлением является веб-сайт GURPS Mentor (рисунок 7, таблица 7). Как и предыдущее приложение, данный инструмент содержит в себе большую базу знаний из базовых книг и дополнительных источников, при этом большая часть контента содержит описание и ссылки на электронные версии книг, а также поддерживается возможность выкладывать листы своих персонажей в общий доступ и просматривать листы персонажей других пользователей.

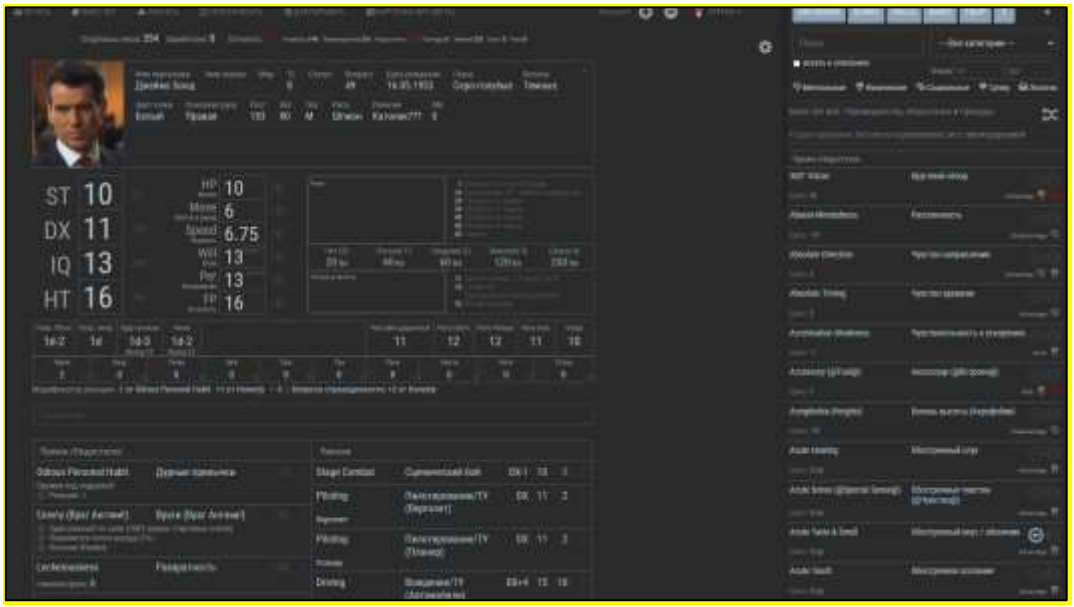


Рисунок 7 – Создание персонажа в GURPS Mentor

Таблица 7 – Преимущества и недостатки GURPS Mentor

Преимущества	Недостатки
Локализован на русский язык	Необходимость подключения к интернету
Большая база знаний	
Автоматический расчёт характеристик	
Регулярные обновления	
Общий доступ и пользовательские библиотеки	
Связь с Discord	
Интуитивно понятный интерфейс	
Простота использования	

Автоматизация настольной ролевой системы GURPS наиболее рациональна по ряду причин: это универсальная система, позволяющая охватить наибольшую целевую аудиторию игроков в настольные ролевые игры; правила данной системы являются комплексными и в дополнительных книгах становятся только сложнее; существующий набор инструментов недостаточен.

Требования к разрабатываемому инструментарию

В рамках данного исследования было решено провести анализ требований к разрабатываемому инструментарию для ответвления GURPS Mass Combat – свода правил для проведения абстрактных массовых боевых столкновений, включающий в себя большое число различных правил, которые создают трудности для ручного расчёта. Далее будут рассмотрены основные требования к системе и интерфейсу разрабатываемого инструментария:

Требования к ПО:

1. ПО должно иметь интуитивно понятный пользовательский интерфейс
2. ПО должно быть разработано под операционные системы Windows и выше.
3. ПО не должно требовать подключения к интернету
4. ПО не должно требовать авторизации пользователя
5. ПО должно быть самостоятельным продуктом и распространяться на бесплатной основе
6. ПО должно создавать боевые единицы (элементы) по правилам GURPS Mas Combat
7. ПО должно хранить информацию об элементах на диске.
8. ПО должно позволять пользователю создание боевого столкновения по правилам GURPS Mass Combat
9. ПО должно автоматически изменять данные армий при манипуляции последней со стороны пользователя или системы
10. ПО должно проводить точный расчет результатов боевого столкновения.
11. ПО должно предоставлять подробную информацию о симуляции пользователю
12. ПО должно обрабатывать исключительные ситуации, возникшие на любом из этапов использования
13. ПО должно хранить информацию о симуляциях сражения по нужде пользователя.
14. ПО должно иметь раздел настроек, в котором пользователь может выбрать дополнительные функции
15. ПО должно поддерживать автосохранение, переключаемое в настройках
16. ПО должно ссылаться на настольную ролевую систему GURPS Mass Combat и на все инфе источники, дополняющие правила данной ролевой системы.

Требование к интерфейсу:

1. Интерфейс должен быть простым и интуитивно понятным для любого пользователя.
2. Все кнопки должны сопровождаться всплывающими подсказками
3. Все окна должны иметь справочную информацию по своему назначению и описанию кнопок.
4. Диалоговые окна об обработке исключительных ситуаций должны понятно объяснять причину ошибки и предлагать её оптимальное решение (если решение возможно, в ином случае самостоятельно разрешить исключительную ситуацию
5. Интерфейс должен осуществлять перемещение между окнами с помощью кнопок.

Выводы

В заключение можно сказать, что исследование по теме данной статьи выявило актуальные проблемы низкой популярности настольных ролевых игр: доступность, сложность и локализация. Большая часть существующих систем либо являются устаревшими, либо не представляют достаточную помощь ведущему или игрокам при проведении игровых сессий. Также исследование указало на наиболее нуждающиеся в автоматизации настольные ролевые системы, а на основе существующих были определены требования к новому разрабатываемому инструментарию. Дальнейшее исследование в этом направлении позволит улучшить качество разрабатываемых систем в сфере настольных ролевых игр, упростить их проведение и повысить заинтересованность целевой аудитории настольных игр на таком жанре, как НРИ.

Литература

1. Игнатьев, Д. Темп роста рынка настольных игр замедлился. – URL: https://www.vedomosti.ru/media/articles/2023/03/15/966529-temp-rosta-rinka-nastolnih-igr-zamedlilsya (дата обращения: 10.05.2024). – Текст : электронный.

2. Челышева, Д. А. Генерация сюжетной составляющей для настольной ролевой игры / Д. А. Челышева, А. В. Ключиков // Программные системы: теория и приложения. – 2023. – Т. 14, № 4(59). – С. 123-140. – DOI 10.25209/2079-3316-2023-14-4-123-140. – EDN GRBQDG.

3. Павлов, М. Ю. Реализация базовых алгоритмов системы, основанной на НРИ, программными методами C# / М. Ю. Павлов, А. В. Боднар // Информатика, управляющие системы, математическое и компьютерное моделирование (ИУСМКМ-2023) : Материалы XIV Международной научно-технической конференции в рамках IX Международного Научного форума Донецкой Народной Республики, Донецк, 24–25 мая 2023 года. – Донецк: Донецкий национальный технический университет, 2023. – С. 144-149. – EDN QHSTDT.

4. Помощник для настольных ролевых игр / М. С. Власенко, Д. К. Фокин, В. С. Якимов, В. С. Андрианов // Проспект свободный - 2020 : Материалы Международной конференции молодых ученых, Красноярск, 20 апреля – 18 2020 года / Отв. за выпуск М.В. Носков. – Красноярск: Сибирский федеральный университет, 2020. – С. 11-12. – EDN ZPJFSQ.

5. Беседин, Д. К. Разработка web-сайта для сопровождения настольной ролевой игры GURPS / Д. К. Беседин // Фундаментальные и прикладные аспекты компьютерных технологий и информационной безопасности : Сборник статей Всероссийской научно-технической конференции, Таганрог, 10–15 апреля 2023 года. – Таганрог: Южный федеральный университет, 2023. – С. 90-92. – EDN XOEMCQ.

6. Mearls, M. D&D Player's Handbook / M. Mearls, J. Crawford; пер. Landor. – Renton : Wizard of the Coast, 2016. – 331 с.

7. Bulmahn, J. Pathfinder Roleplaying Game: Core Rulebook / J. Bulmahn. – Paizo Inc., 2009. – 464 с.

8. Petersen, S. Call of Cthulhu Investigators Handbook (Call of Cthulhu Roleplaying) / S. Petersen, L. Willis, P. Fricker; ред. M. Mason. – Chaosium, 2016. – 288 с.

9. Johnson, S. Basic Roleplaying (2nd Edition) / S. Johnson, J. Durall. – Chaosium, 2008. – 399 с.

10. Jackson, S. GURPS Basic Set: Characters, Fourth Edition / S. Jackson, S. Punch, D. Pulver; ред. Andrew Hackard. – Steve Jackson Games, 2016. – 336 с.

Айдин С.А., Боднар А.В. Применение информационных технологий в сфере настольных ролевых игр. *В статье рассматриваются основные понятия настольных ролевых игр, проводится анализ наиболее распространённых настольных ролевых систем и оценивается рациональность их автоматизации. Также анализируются существующие наборы инструментов и формируются требования к разрабатываемому ПО. Направлением дальнейших исследований является улучшение качества разрабатываемых систем в сфере настольных ролевых игр, что позволит упростить их проведение и повысить заинтересованность целевой аудитории настольных игр на таком жанре.*

Ключевые слова: *НРИ, ролевые системы, инструментарий, автоматизация, GURPS, Mass Combat.*

Aidin S.A., Bodnar A.V. The use of information technology in the field of tabletop role-playing games. *The article discusses the basic concepts of tabletop role-playing games, analyzes the most common tabletop role-playing systems and evaluates the rationality of their automation. The existing toolkits are also analyzed and requirements for the software being developed are formed. The direction of further research is to improve the quality of the systems being developed in the field of tabletop role-playing games, which will simplify their implementation and increase the interest of the target audience of board games in this genre.*

Keywords: *TTRPG, role-playing systems, tools, automation, GURPS, Mass Combat*

Статья поступила в редакцию 14.04.2024
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 004.853

Применение подхода «архитектура как код» при формировании крупных экосистем

К. А. Терещенко, Р. В. Мальчева
Донецкий национальный технический университет
E-mail: raisa.malcheva@yandex.ru

Аннотация
Рассмотрены основные методологии реализации процесса развития проекта. Сформулированы проблемы контроля и развития системной архитектуры в рамках основных рассматриваемых методологий. Рассмотрен новый подход к реализации артефактов системной архитектуры в процессе проектной работы. В результате анализа сделан вывод, что для крупных систем внедрение отдельной команды и набор соответствующих специалистов является менее ресурсозатратной задачей, чем внедрение в каждую команду отдельного специалиста по архитектуре или разрешение постоянно всплывающих инцидентов.

Введение

При формировании крупных экосистем актуальной проблемой является проектирование их эффективной архитектуры с учетом таких ключевых требований как масштабируемость, открытость, неоднородность, безопасность, доступность, и производительность [1-5]. В [6] выполнен анализ особенностей и области применения шаблонов системной архитектуры с точки зрения эффективности их применения в развернутых экосистемах.

Целью данной статьи является анализ основных методологий реализации процесса развития проекта и рассмотрение нового подхода к реализации артефактов системной архитектуры в процессе проектной работы.

Общая постановка проблемы

Формирование крупной IT экосистемы - не тривиальная задача. Для того, чтобы привлечь пользователей, современные компании пытаются не просто предоставить им набор разнообразных обособленных продуктов, а сформировать совокупный ландшафт интегрированных друг с другом решений. Это делается с целью повышения степени автоматизации сопряженных друг с другом процессов, улучшения отчетности, снижения нагрузки на пользователя и сохранения его в экономическом контуре компании. В итоге рождается задача контроля, необходимая для выполнения условий функционирования, масштабирования и развития множества связанных между собой проектов. Также возникает набор новых ролей, одной из которых является роль архитектора систем.

Системный архитектор — это специалист, который отвечает за создание и разработку общей структуры и дизайна информационной системы или программного продукта. Он определяет ключевые компоненты системы, их взаимодействие и общую архитектуру, учитывая требования заказчика, бизнес-процессы и технические аспекты. Роль системного архитектора включает определение архитектурных решений, разработку технических спецификаций, управление техническими рисками и обеспечение соответствия архитектуры стандартам и требованиям проекта.

Ресурсы, выделяемые архитекторам на работу с проектом, тяжело планировать и балансировать. В крупных компаниях, где существует значительное число команд, постоянное согласование изменений с архитектором может существенно замедлить процесс разработки. В свою очередь, увеличение числа архитекторов, напротив, может создать ситуацию простаивания ресурсов, так как архитектор участвует лишь в некоторых этапах создания системы (планирование, согласование архитектурно значимых изменений).

В отличие от процесса разработки, где все построено на коде, архитектура системы требует создания графических артефактов, для которых используются визуальные редакторы.

Объединение версий диаграмм, актуализация данных, плановые изменения расположенных в документации решений часто приводит к значительным затратам времени, так как даже определение расположения всех связанных с проектом артефактов в масштабных системах становится комплексной и развернутой задачей. Тратятся значительные ресурсы на механическую и бюрократическую деятельность, что может стать причиной возникновения неактуальной документации, замедления процессов согласования, расхождения в понимании концептов реализации

взаимодействия систем и сервисов в разных командах этих систем, и, как следствие, приведет к появлению ошибок в интеграционном взаимодействии и возникновению инцидентов.

Анализ существующих методологий

Существуют разные методологи для решения вышеописанной проблемы. Использование одного из популярных концептов ведения разработки потенциально позволяет организовывать процесс более эффективным образом, что решает множество проблем, таких как возникновение неактуальной документации, расхождения в системных контрактах взаимодействия и других, связанных с реализацией системной архитектуры.

Можно выделить несколько основных металогических подходов к разработке.

Agile-методологии (рисунок 1) - семейство методологий управления проектами и разработки программного обеспечения, которые основаны на принципах Agile Manifesto [7]. Они предполагают итеративное и инкрементальное развитие продукта, постоянное взаимодействие с заказчиком, гибкое реагирование на изменения в требованиях и фокус на командной работе. Применение данного шаблона могло бы позволить сократить объем концентрации ответственности на системном архитекторе, распределяя ее по проектным командам, которые берут на себя часть функций в рамках производственного цикла.

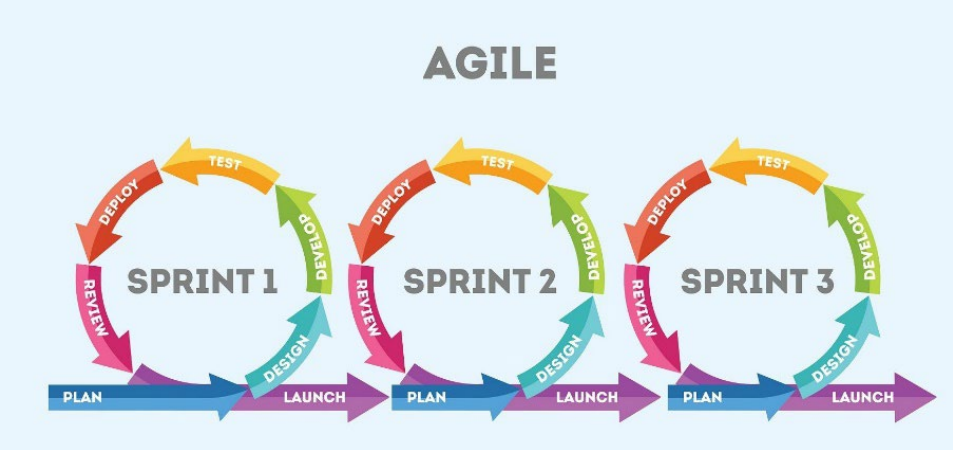


Рисунок 1 – Agile-методологии

Рассмотрим некоторые из проблем применения подхода в крупных компаниях.

Необходимость стандартизации. В крупных корпорациях часто имеются строгие архитектурные стандарты и процедуры, которым должны следовать все проекты. Это касается как применяемых технологий с целью унификации подхода к мониторингу и обмену данными, например, внедрения определенных брокеров сообщений или стандартизированных API интерфейсов, так и формирования конкретизированных архитектурных шаблонов, предназначенных, например, для ограничения хранения критически важных данных в полном множестве проектов. Agile-методологии, напротив, поощряют гибкость и адаптацию к изменяющимся требованиям. Это может привести к конфликтам со стандартами и процедурами корпорации и дальнейшим проблемам.

Масштабирование. Agile-методологии изначально были разработаны для небольших команд, что может вызвать сложности для масштабирования их до уровня системной архитектуры крупной корпорации. При значительном числе штаба разбиение на малые команды может затруднить коммуникацию, а расширение состава команд стать препятствием к реализации уже самого манифеста.

Waterfall-модель (рисунок 2) [8]. Это традиционный подход к управлению проектами, который предполагает последовательное выполнение этапов проекта: анализ требований, проектирование, реализацию, тестирование и внедрение. Этот подход хорошо подходит для проектов с четко определенными требованиями и стабильной средой разработки. Несмотря на то, что waterfall модель в определенной степени решает проблемы Agile методологии, отсутствие гибкости порождает ряд новых.

*Высокая стоимость ошибок. Е*сли ошибки были допущены на ранних стадиях проекта, то исправление их может занять много времени и ресурсов, так как waterfall не предполагает возвращение к ранним этапам. Это особенно болезненно для крупных и развернутых систем, где цена простоя крайне высока.

Проблемы с коммуникацией. При использовании Waterfall-модели необходимо четко определить все требования к проекту на начальном этапе.

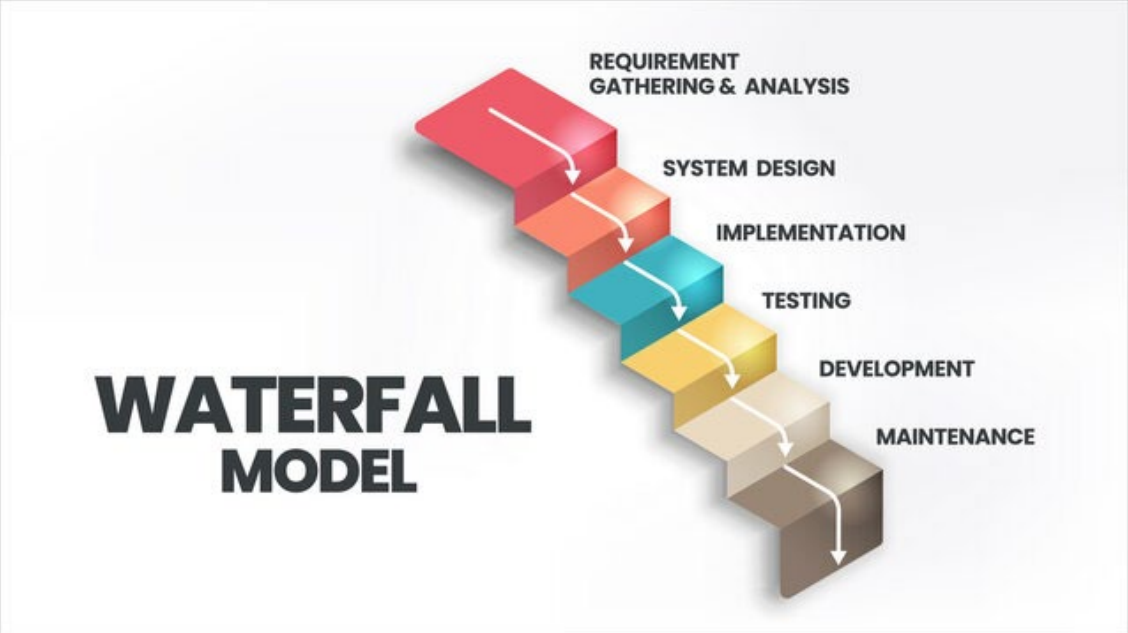


Рисунок 2 – Waterfall-модель

Однако, в крупных проектах часто возникают сложности с коммуникацией между различными отделами и заинтересованными сторонами, что нередко приводит к рассинхронизации данных, нарушению системных контрактов и запозданию в локализации ошибок.

Model-driven engineering (MDE) [9]. Данная методология в области разработки программного обеспечения основывается на использовании моделей для проектирования и создания систем. В своей структуре она включает формирование необходимых артефактов системной архитектуры. Более того, в MDE модели играют центральную роль, представляя систему и ее компоненты на различных уровнях абстракции.

Разработчики используют модели для автоматизации процессов создания кода, тестирования, документирования и сопровождения ПО, что может решить ряд проблем с коммуникацией, так как мутабельность моделей автоматизируема и позволяет отслеживать все необходимые изменения в взаимосвязанных технически артефактах документации.

Тем не менее данный подход тяжело применим, так как:

- требует использования или разработки специализированных инструментов автоматизации формирования артефактов;
- предлагает лишь набор стандартизированных принципов, а не конкретные решения, что в контексте применения автоматизации и моделирования, по сути, ставит перед проектом задачу реализации подхода.

Это привело к разработки его модификаций.

Архитектура как код

Данная архитектура фактически является локализацией **MDE,** определяющей более конкретный процесс создания архитектурных артефактов системы, который потенциально может быть расширен на весь процесс разработки.

Концептуально «архитектура как код» сопоставляет процесс формирования моделей, системных контрактов и другой документации с процессами внедрения изменений в рабочий код компании [10].

Артефакты создаются путем написания кода с использованием определенных инструментов, позволяющих это делать, например, PlantUML.

Затем процесс фиксируется через инструмент контроля версий, например Git, в общем «репозитории» проекта.

В случае внесения изменений, автор создаёт отдельную «ветку», в которую вносит необходимые доработки (рисунок 3) и отправляет ее «на слияние» (рисунок 4) с корневым проектом.

Ответственные за архитектуру проверяют доработки и согласовывают их, либо возвращают автору с комментариями, какие дополнительные изменения требуется произвести.

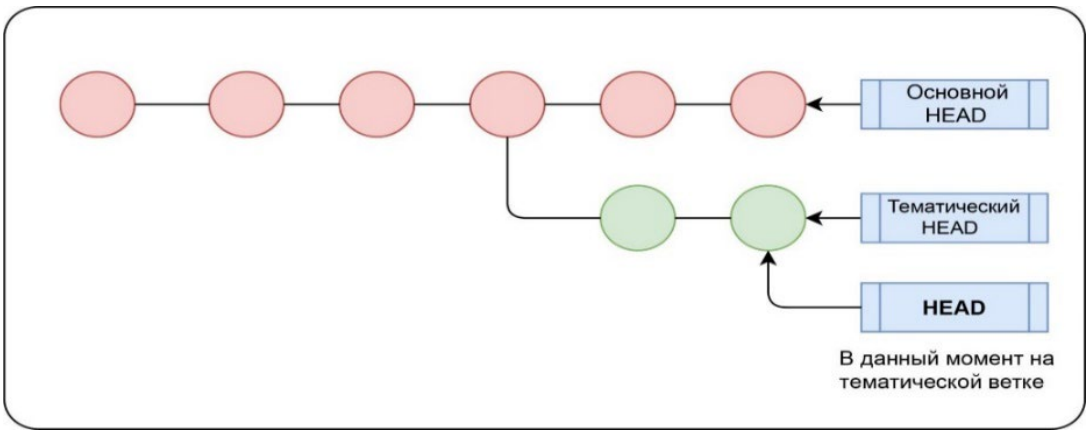


Рисунок 3 - Создание тематической ветки (HEAD)

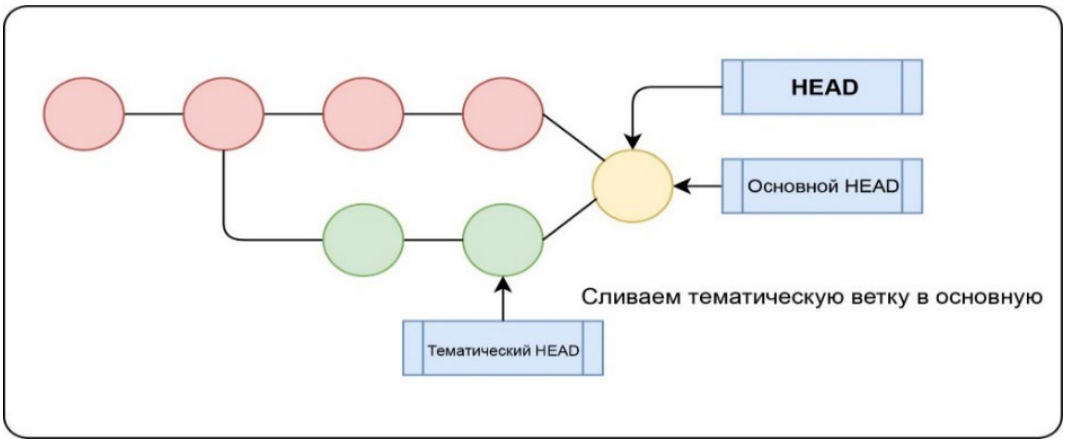


Рисунок 4 - Слияние веток

Помимо внутренней ядра проекта есть корпоративный центральный репозиторий (рисунок 5), который содержит концептуальную архитектуру, доступную всем для ознакомления с планами развития.

Изменения в репозиторий вносятся также через «Pull requests», а визуализация происходит средствами какого-либо инструмента, например, DocHub.

Далее изменения рассматривают управляющие проектами. Это способствует обмену опытом, установке эффективного взаимодействия и общему пониманию архитектурных решений.

Описание текущей архитектуры хранится в отдельных командах, что дает им возможность автономно принимать быстрые решения.

Централизованный репозиторий размещен рядом с командами, позволяя им оперативно вносить обоснованные изменения в продукт, и, в случае необходимости, фиксирует отклонения как инциденты.

Выводы

Реализация концепции «архитектура как код» имеет преимущества, которые частично разрешают противоречия, возникающие при применении популярных методологий Agile и Waterfall, а также дают локализацию и конкретику в применении методологии MDE.

Применение подхода накладывает определенные требования, например, дополнительными компетенциями на соответствующих участников процесса, использование отдельной системы для контроля системной архитектуры и другие.

В результате анализа сделан вывод, что для крупных систем внедрение отдельной команды и набор соответствующих специалистов является менее ресурсозатратной задачей, чем внедрение в каждую команду отдельного специалиста по архитектуре или разрешение постоянно всплывающих инцидентов.

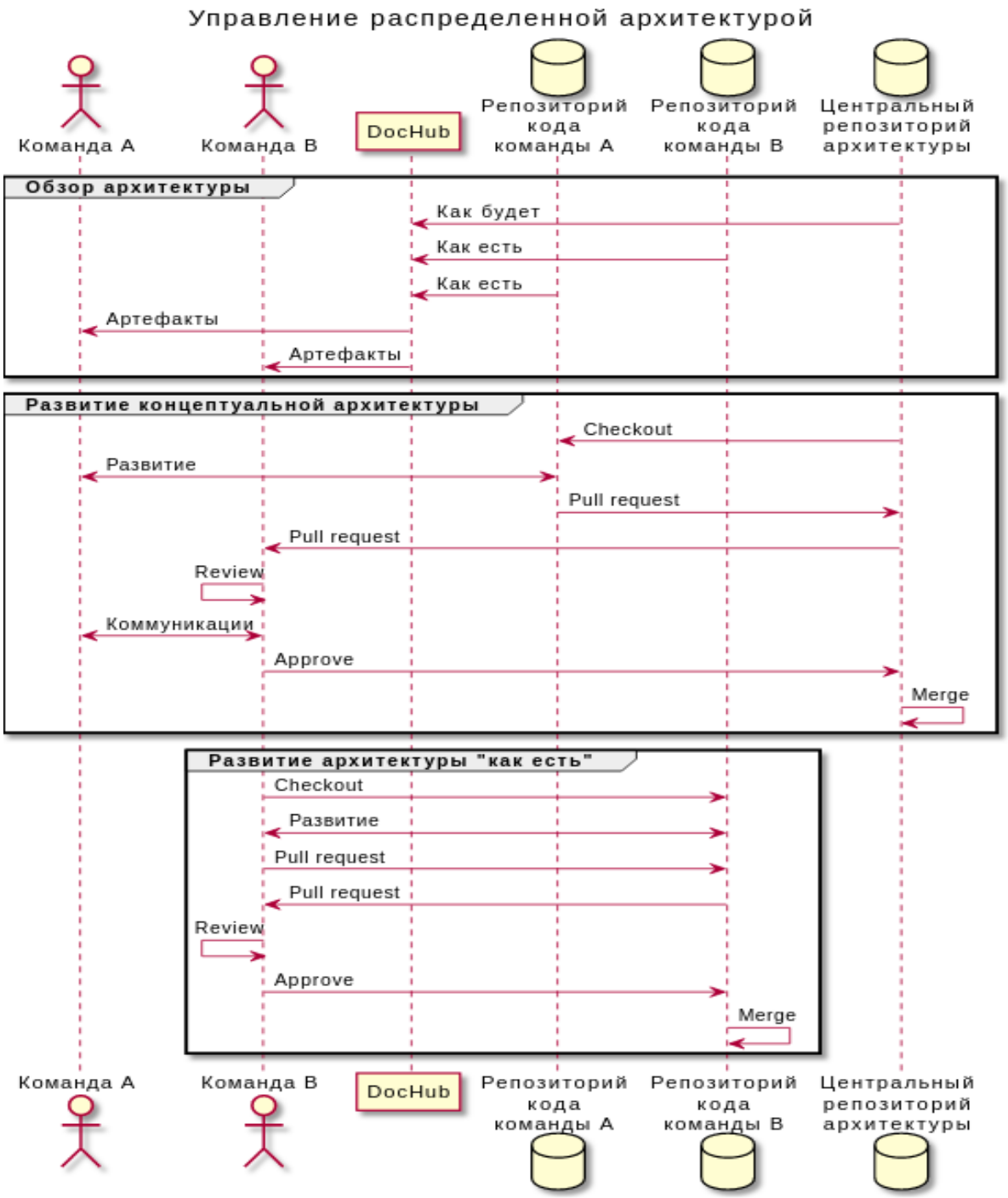


Рисунок 5 – Управление распределенной архитектурой

Литература

1. Мальчева, Р. В. Компьютерные технологии – основа цифровой экономики // Бизнес-инжиниринг сложных систем: модели, технологии, инновации. Сборник материалов III международной научно-практической конференции. – Донецк: ДОННТУ, 2018. - С. 102-105.

2. Паньшин, Б. Цифровая экономика: особенности и тенденции развития [Электронный ресурс] / Б. Паньшин. – Режим доступа: https://cyberleninka.ru/article/n/tsifrovaya-ekonomika-osobennosti-i-tendentsii-razvitiya

3. Мальчева, Р. В. Проектирование архитектуры системы электронных денег / Р. В. Мальчева, К. А. Терещенко // Информатика и кибернетика. – Донецк: ДонНТУ, 2023. - № 1(31). – С. 23-29.

4. Терещенко, К. А. Проектирование распределенной архитектуры компьютерной сети банка / К. А. Терещенко, Р. В. Мальчева // Информационное пространство Донбасса: проблемы и перспективы : материалы V Респ. С междунар. Участием науч.-практ. Конф., 27 окт.

2022 г. – Донецк : ГОУ ВПО «ДонНУЭТ», 2022. – С. 131 -134.

5. Терещенко, К. А. Анализ особенностей функционирования и развития объектов и процессов компьютерной сети банка / К. А. Терещенко, Р. В. Мальчева // Информатика, управляющие системы, математическое и компьютерное моделирование (ИУСМКМ-2022): Материалы XIII Международной научно-технической конференции в рамках VIII Международного Научного форума Донецкой Народной Республики. Донецк, 2022. – Донецк: ДонНТУ, 2022. – С.392-295.

6. Терещенко К. А. Анализ специфики и области применения архитектурных шаблонов в развернутых экосистемах / К. А. Терещенко, Р. В. Мальчева // Информатика и кибернетика. – Донецк: ДонНТУ, 2023. - № 4(34). – С. 49-59.

7. Agile: что это такое и где используется, принципы методологии [Электронный ресурс]. – Режим доступа: https://practicum.yandex.ru/blog/metodology-agile/

8. Барулина, В. В. Сравнительный анализ гибкой и каскадной методологии разработки программного обеспечения [Электронныйресурс] / В. В. Барулина. – Режим доступа: https://cyberleninka.ru/article/n/sravnitelnyy-analiz-gibkoy-i-kaskadnoy-metodologii-razrabotki-programmnogo-obespecheniya/viewer

9. Alberto Rodrigues da Silva. Model-Driven Engineering: A Survey Supported by a Unified Conceptual Model [Электронный ресурс] / INESC-ID / Instituto Superior Técnico, Universidade de Lisboa. – Режим доступа: http://isg.inesc-id.pt/alb/static/papers/2015/r13-as-2015-COMLAN-v2.1b.pdf

10. Brown, S. A minimal approach to software architecture documentation [Электронный ресурс] / S. Brown. – Режим доступа: https://dev.to/simonbrown/a-minimal-approach-to-software-architecture-documentation-4k6k

Терещенко К.А., Мальчева Р.В. Применение подхода «архитектура как код» при формировании крупных экосистем. *Рассмотрены основные методологии реализации процесса развития проекта. Сформулированы проблемы контроля и развития системной архитектуры в рамках основных рассматриваемых методологий. Рассмотрен новый подход к реализации артефактов системной архитектуры в процессе проектной работы. В результате анализа сделан вывод, что для крупных систем внедрение отдельной команды и набор соответствующих специалистов является менее ресурсозатратной задачей, чем внедрение в каждую команду отдельного специалиста по архитектуре или разрешение постоянно всплывающих инцидентов.*

Ключевые слова: *экоситема, шаблон, архитектура, критерии, система оценок*

Tereshchenko K.A., Malcheva R.V. The application of the "architecture as code" approach in the formation of large ecosystems. *The main methodologies for the implementation of the project development process are considered. The problems of control and development of the system architecture within the framework of the main methodologies under consideration are formulated. A new approach to the implementation of system architecture artifacts in the process of project work is considered. As a result of the analysis, it was concluded that for large systems, the introduction of a separate team and the recruitment of appropriate specialists is a less resource-intensive task than the introduction of a separate architecture specialist into each team or the resolution of constantly emerging incidents.*

Key words: *ecosystem, pattern, architecture, criteria, evaluation system*

Статья поступила в редакцию 15.04.2024
Рекомендована к публикации профессором Зори С. А.

УДК 004.94

Влияние периодической и экспоненциальной составляющих развития цифровых технологий на процессы и явления в компьютерном моделировании

Н. А. Бездетный, С. А. Зори
Донецкий национальный технический университет,
nekooolay@mail.ru, SPIN-код: 2472-1006
ik.ivt.rec@mail.ru, OrcID: 0000-0003-4018-234X, SPIN-код: 3565-6330

Аннотация
В данной статье анализируются экспоненциальные зависимости развития компьютерного моделирования, а также особое внимание уделяется периодическим закономерностям и их связям с научными открытиями и технологическими прорывами на протяжении всей истории моделирования, начиная с древних цивилизаций. Показана ключевая роль компьютерного моделирования в различных сферах деятельности, включая науку, инженерию, медицину, экономику. Рассматривается влияние моделирования на повышение точности прогнозирования, оптимизацию процессов, развитие новых технологий и ускорение научного прогресса.

Введение

Моделирование и симуляция процессов составляет неотъемлемую часть человеческой деятельности — от имитации простой музыки до комплексных систем запуска спутников в космос. Даже история науки и техники — это история формирования и развития моделируемых явлений, процессов и объектов.

Модель — это искусственно создаваемый объект, заменяющий некоторую сущность реального мира и воспроизводящий ограниченное число его свойств. Процесс построения модели называют моделированием. Моделирование — это метод познания действительности, используемый различными науками [1].

Моделирование является важным инструментом в научных и инженерных исследованиях, позволяющим создавать упрощенные абстрактные представления реальных систем. Оно позволяет нам лучше понять и предсказывать поведение сложных систем, а также проводить различные эксперименты и анализировать результаты без необходимости проведения физических опытов [2]. С началом эпохи цифровизации и приходом компьютеров в нашу жизнь, возникла возможность создавать модели различных систем и процессов с помощью программного обеспечения. Это привело к развитию компьютерного моделирования – метода анализа и прогнозирования поведения объектов в виртуальной среде.

Компьютерное моделирование позволяет создавать абстрактные представления реальных систем, исследовать их свойства и взаимодействия без необходимости проведения физических экспериментов. Например, с помощью компьютерного моделирования можно предсказывать погодные условия, анализировать движение транспортных потоков, проектировать новые материалы или оценивать эффективность лекарственных препаратов.

Таким образом, компьютерное моделирование играет важную роль в науке, инженерии, медицине и других областях, предоставляя исследователям мощный инструмент для системного анализа и улучшения различных систем и процессов.

Влияние периодической составляющей

Моделирование имеет долгую историю, начиная с древних времен. Можно выделить такие периодические зависимости на протяжении всей истории (рис. 1) [3, 4, 5]:

В Древнем Египте (3000 г. до н.э. – 30 г. н.э.) использовали модели для планирования и строительства пирамид, храмов и других сооружений. Развитие методов и материалов для создания моделей, созданных из дерева, глины или камня, позволяло лучше визуализировать и проверять конструкции перед фактическим строительством.

В Древней Греции (800 г. до н.э. – 600 г. н.э.) использовали моделирования в математике и геометрии для изучения геометрических форм и теорем. Разработка моделей помогла греческим математикам (Пифагор, Евклид и Архимед) лучше понять и доказать различные математические концепции.

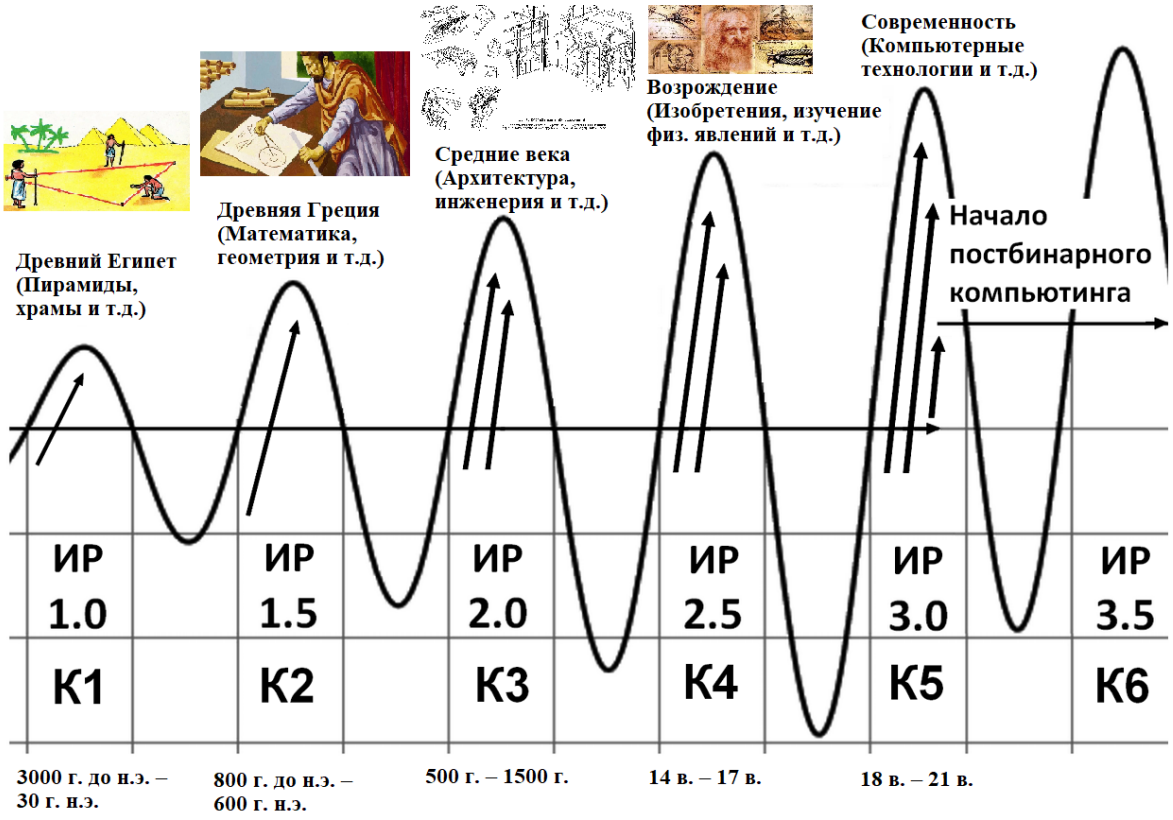


Рисунок 1 – Периодические зависимости развития моделирования

В Средние века (500 г. – 1500 г.) использовали модели в архитектуре и инженерии для проверки прочности и стабильности зданий, мостов и других сооружений. Создание более точных и реалистичных моделей, созданных из дерева, камня или металла, помогло улучшить планирование и строительство различных сооружений.

Во эпоху Возрождения (14 в. – 17 в.) происходило систематическое и научное использование моделирования для изучения физических явлений и разработки новых изобретений (Леонардо да Винчи, Никола Тесла). При создании более сложных и точных моделей развивались методы моделирования, что способствовало улучшению проектирования и эффективности различных устройств и механизмов.

В Современности (18 в. – настоящее время) происходит широкое применение компьютерной технологии и математических методов в моделировании. Современное моделирование стало более точным, быстрым и мощным благодаря развитию компьютеров и алгоритмов, что привело к расширению областей применения моделирования в науке, инженерии, экономике, медицине и других отраслях для анализа данных, прогнозирования результатов, оптимизации процессов и принятия решений.

Влияние экспоненциальной составляющей

Экспоненциальная составляющая развития цифровых технологий напрямую влияла на процессы и явления компьютерного моделирования [4, 5]. На приведенной ниже диаграмме выделены периоды, где происходили резкие скачки экспоненциального развития компьютерного моделирования (рис. 2).

В середине 20-го века происходит развитие алгоритмов и методов численного моделирования. Создание первых электронных компьютеров, таких как ENIAC, Colossus, МЭСМ, БЭСМ, способствует быстрому увеличению вычислительной мощности, что позволяет разработать более сложные и точные алгоритмы для моделирования различных физических и инженерных явлений, а также их активное применение в инженерной практике и научных исследованиях (рис. 3).

В 70-80 года происходит распространение ПК и развитие программного обеспечения. Появление персональных компьютеров и специализированного программного обеспечения увеличило доступность средств моделирования для широкой аудитории, что стимулировало развитие индустрии и исследований в области компьютерного моделирования (рис. 4).

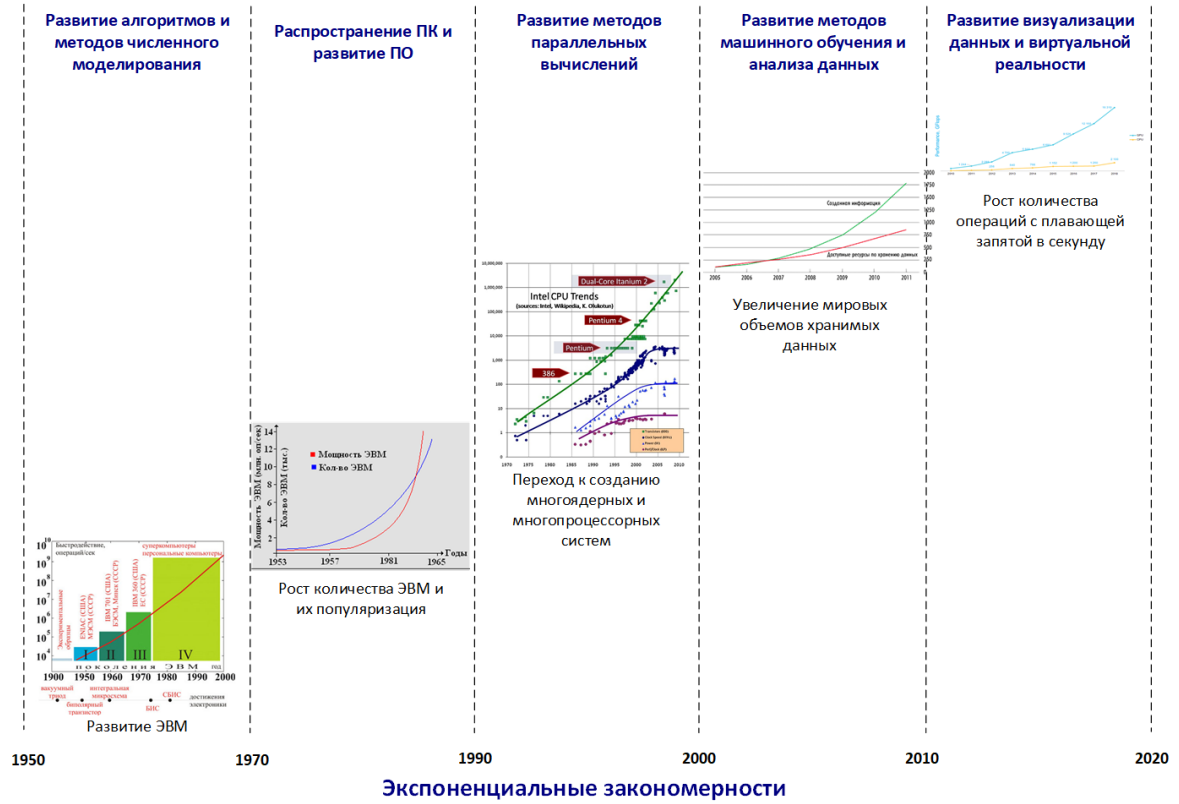


Рисунок 2 - Экспоненциальные зависимости развития компьютерного моделирования

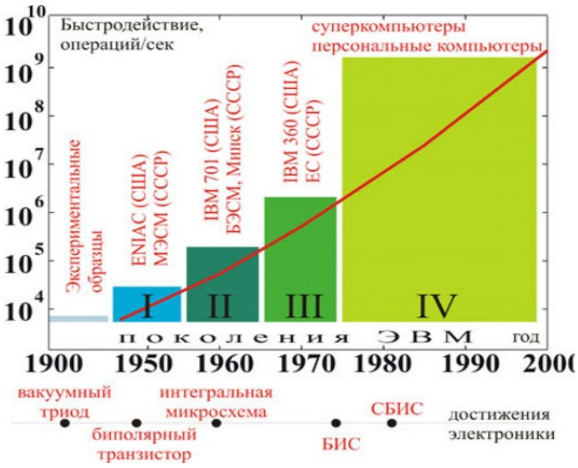


Рисунок 3 - График развития ЭВМ

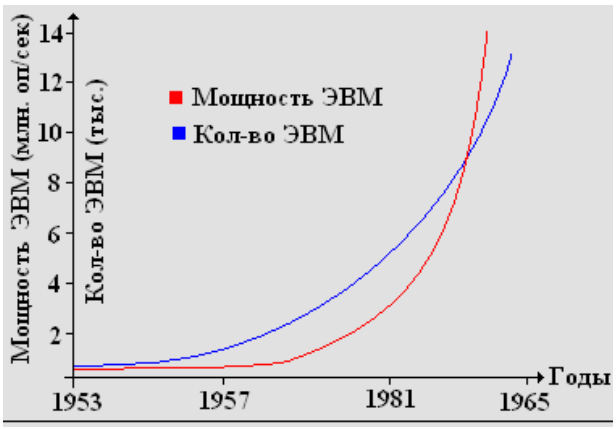


Рисунок 4 - Тенденции роста количества и мощности ЭВМ

В 90-е происходит развитие методов параллельных архитектур и вычислительных технологий. Когда требовалось повысить производительность процессоров, основной стратегией было увеличение их тактовой частоты - скорости работы. Однако по мере того, как процессоры становились все более быстрыми, возникли физические ограничения, такие как ограничения на энергопотребление и тепловыделение. Это ограничивало возможности дальнейшего увеличения частоты без перегрева или повышения энергопотребления до неприемлемых уровней (рис. 5).

Вместо того, чтобы продолжать идти по пути увеличения частоты, инженеры обратили свой взгляд на распараллеливание вычислений. Распараллеливание позволяет выполнять несколько задач одновременно, используя для этого несколько процессорных ядер или процессоров. Это подходит для многих современных приложений, так как многие из них могут быть разделены на более мелкие части, которые могут выполняться параллельно.

Таким образом, вместо того, чтобы стараться делать процессоры все быстрее и быстрее, индустрия перешла к созданию многоядерных и многопроцессорных систем.

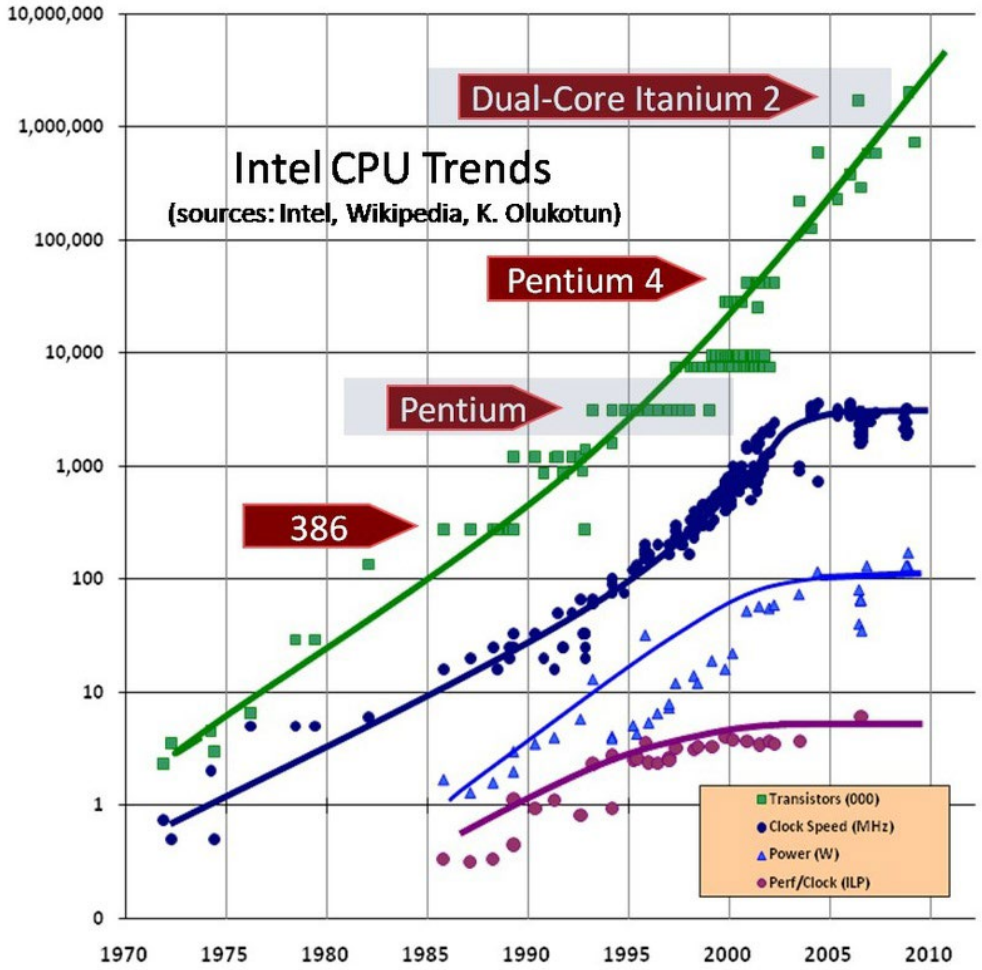


Рисунок 5 – Число транзисторов, тактовая частота, энергопотребление и степень параллелизма на уровне инструкций

Это позволило повышать общую производительность, используя параллельную обработку, вместо простого увеличения скорости одного процессора [6].

В начале 21 века объем доступных данных для анализа и обработки вырос в несколько десятков раз благодаря развитию интернета, цифровых устройств и технологий сбора информации**.** Развитие методов машинного обучения, глубокого обучения и анализа больших данных, позволило создавать более точные и адаптивные модели, а также использовать большие объемы данных для обучения моделей (рис. 6) [7].

Рисунок 6 – Объем создаваемой информации в мире (в эксабайтах)

Начиная с 2010 года, развитие графических технологий и виртуальной реальности обусловлено резким увеличением производительности видеокарт, измеряемой во flops (количество операций с плавающей запятой, которое может выполнить компьютер за единицу времени) [8]. Видеокарты стали значительно превосходить процессоры по этому показателю, способствуя развитию компьютерного моделирования и обработки графики. Этот прогресс расширяет возможности создания более реалистичных и информативных визуализаций данных, что в свою очередь улучшает восприятие результатов моделирования и облегчает принятие решений (рис. 7) [9].

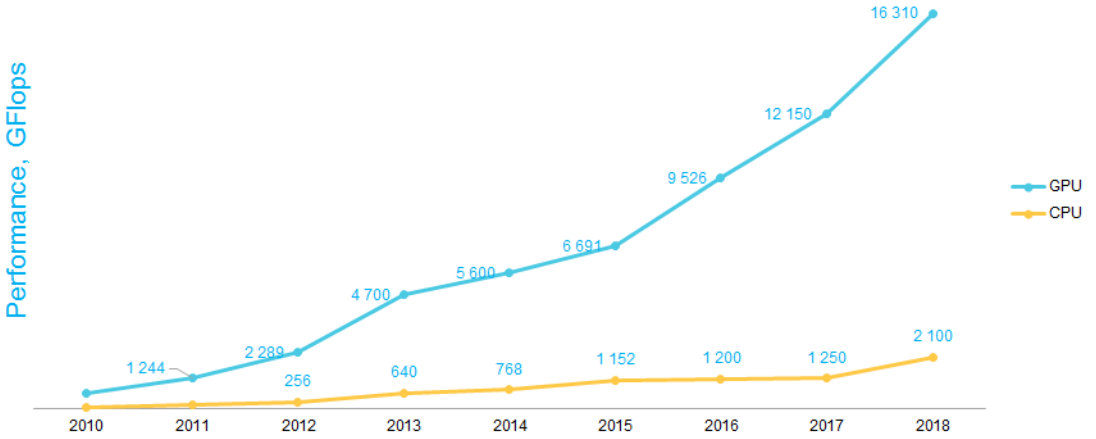


Рисунок 7 – Сравнение роста GFlops для GPU и CPU

Также следует отметить экспоненциальный рост научного интереса к теме компьютерного моделирования. За последние 20 лет количество публикаций по данной теме возросло в 8 раз, что указывает на значительное увеличение активности и изучения этой темы в научном сообществе. Этот тренд, может свидетельствовать о растущем влиянии и значимости компьютерного моделирования в науке и технологиях (рис. 8).

Рисунок 8 – Гистограмма количества публикаций, связанных с компьютерным моделированием

Таким образом, история развития моделирования – это длительный процесс становления от простых физических моделей до сложных компьютерных симуляций, демонстрирующий постоянное стремление человечества к пониманию окружающего мира. Технологический рост, экспоненциальные скачки вычислительных мощностей, а также развитие программного обеспечения стали катализаторами современного прогресса, превратив моделирование из узкоспециализированного инструмента в универсальный метод познания, применяемый практически во всех сферах жизни.

Если представить мир, где не было бы такого бурного развития технологий –

моделирование оставалось бы уделом узкого круга специалистов, а его возможности были бы ограничены. Мы бы не смогли решать сложные задачи, такие как прогнозирование погоды, разработка новых материалов или моделирование поведения экономических систем. Развитие моделирования привело к настоящей революции во многих областях науки, техники и общества в целом. Мы можем проектировать более безопасные автомобили, предсказывать стихийные бедствия и даже моделировать эволюцию Вселенной. Это открывает перед нами невероятные возможности для улучшения качества жизни, решения глобальных проблем и расширения горизонтов нашего познания.

Понимание закономерностей развития цифровых технологий, их периодичности и экспоненциального роста, является ключевым фактором для успешного развития компьютерного моделирования. Анализ исторических циклов позволяет прогнозировать будущие тенденции и определять перспективные направления исследований и разработок. Это помогает эффективно распределять ресурсы, инвестируя в развитие технологий и компетенций, которые будут востребованы в будущем.

Экспоненциальный рост цифровых технологий открывает возможности для создания более сложных и точных моделей, решения ранее недоступных задач и получения новых знаний. Это ускоряет научный прогресс и ведет к технологическим прорывам, которые меняют нашу жизнь. Однако экспоненциальный рост также требует гибкости и готовности к постоянному обучению и адаптации. Нужно быть готовым к быстрым изменениям и уметь приспосабливаться к новым условиям. В целом, понимание закономерностей развития цифровых технологий позволяет нам эффективнее планировать будущее, использовать возможности и быть готовыми к проблемам, которые несет с собой стремительный прогресс.

Выводы

В данной работе представлен всесторонний анализ эволюции компьютерного моделирования, от его зарождения до современных тенденций, основанных на машинном обучении и виртуальной реальности. Выделены периодические зависимости и экспоненциальные скачки роста компьютерного моделирования, оказавших значительное влияние на науку, технологии и различные сферы деятельности.

Моделирование является важным инструментом в науке и познании, позволяющим создавать упрощенные представления о сложных системах. Оно позволяет нам лучше понять и объяснить явления, предсказывать их поведение и принимать обоснованные решения. Методы моделирования применяются в различных областях, таких как физика, экономика и т.д. Однако, необходимо учитывать ограничения и критику моделирования, так как модели всегда являются упрощенными представлениями и могут быть неполными или ошибочными.

Исследование эволюции компьютерного моделирования, от его истоков до современных тенденций, позволяет сделать ряд важных выводов.

Во-первых, моделирование является незаменимым инструментом научного познания и технологического прогресса.

Во-вторых, развитие компьютерного моделирования характеризуется периодическими зависимостями и экспоненциальными скачками роста, обусловленными развитием цифровых технологий.

В-третьих, понимание этих закономерностей позволяет нам эффективнее планировать будущее, развивать необходимые компетенции и готовиться к новым проблемам.

В-четвертых, несмотря на ограничения и критику, компьютерное моделирование играет ключевую роль в развитии науки, технологий и общества в целом.

Литература

1. Актуальность моделирования [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://lektsii.org/3-13711.html - Загл. с экрана. (дата обращения: 04.04.24).

2. Моделирование: ключевые понятия, методы и применение в различных областях [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://nauchniestati.ru/spravka/modelirovanie-kak-metod-poznaniya/?ysclid=ltvbiledpr688083100 – Загл. с экрана. (дата обращения: 04.04.24).

3. Эра цифровых технологий: История развития компьютеров и интернета [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://dzen.ru/a/ZUKmcD-x1gfCzIga – Загл. с экрана. (дата обращения: 04.04.24).

4. Аноприенко, А. Я. Нооритмы: Модели синхронизации человека и космоса [Текст] / А. Я. Аноприенко // Издание 3-е, дополненное. – Донецк: УНИТЕХ, 2020. –378 с., ил.

5. Аноприенко, А. Я. Введение в постбинарный компьютинг. Арифметико-логические основы и программно-аппаратная реализация [Текст] / А. Я. Аноприенко, С. В. Иваница // Донецк: ДонНТУ, УНИТЕХ, 2017 — 308 с.

6. Нечаевский, А. В. История развития компьютерного имитационного моделирования

[Текст] / А. В. Нечаевский // Системный анализ в науке и образовании, 2013. - Москва: Мупоч "Дубна". – Вып. №2. - 15 с.

7. Соснин, В. В. Введение в параллельные вычисления : учебное пособие [Текст] / В. В. Соснин, П. В. Балакшин. – СПб.: Университет ИТМО, 2015. – 51 с.

8. От пионеров до прорывов: История развития машинного обучения [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://dzen.ru/a/ZHfSPN-U6xeL41QP - Загл. с экрана. (дата обращения: 24.04.24).

9. Хронология: как развивалась виртуальная, дополненная и смешанная реальности [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://vc.ru/future/44433-hronologiya-kak-razvivalas-virtualnaya-dopolnennaya-i-smeshannaya-realnosti?ysclid=ltvftei2r713842237 - Загл. с экрана. (дата обращения: 24.04.24).

10. Вычисления на GPU – зачем, когда и как. Плюс немного тестов [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://habr.com/ru/companies/dbtc/articles/498374 / - Загл. с экрана. (дата обращения: 24.04.24).

Бездетный Н. А., Зори С. А. Влияние периодической и экспоненциальной составляющих развития цифровых технологий на процессы и явления в компьютерном моделировании. *В данной статье анализируются экспоненциальные зависимости развития компьютерного моделирования, а также особое внимание уделяется периодическим закономерностям и их связям с научными открытиями и технологическими прорывами на протяжении всей истории моделирования, начиная с древних цивилизаций. Показана ключевая роль компьютерного моделирования в различных сферах деятельности, включая науку, инженерию, медицину, экономику. Рассматривается влияние моделирования на повышение точности прогнозирования, оптимизацию процессов, развитие новых технологий и ускорение научного прогресса.*

Ключевые слова: *компьютерное моделирование, история моделирования, экспоненциальный рост, периодические зависимости, визуализация.*

Bezdetniy N.A., Zori S.A. The impact of periodic and exponential components of the development of digital technologies on processes and phenomena in computer modeling. *This paper analyses the exponential dependencies of computer simulation development, and pays special attention to periodic patterns and their connections with scientific discoveries and technological breakthroughs throughout the history of simulation, starting from ancient civilizations. The key role of computer modelling in a variety of fields including science, engineering, medicine, and economics is shown. The impact of modelling on improving the accuracy of forecasting, optimizing processes, developing new technologies and accelerating scientific progress is examined.*

Keywords: *computer modelling, history of modelling, exponential growth, periodic dependencies, visualization.*

Статья поступила в редакцию 15.04.2024
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 004.436.4:004.8

Принципы IDEF как средства реализации мета-эвристической оболочки

Д.А. Филипишин, А.В. Григорьев
Донецкий национальный технический университет, г.Донецк
domaco@mail.ru, grigorievalvl@gmail.com

Аннотация
С целью повышения функциональности существующих концепций нотации информационных систем, которые не позволяют на текущий момент замкнуть весь цикл проектирования программных систем на себе выполнен анализ существующих стандартов нотации IDEF, как наиболее полно охватывающий все виды проектирования программного обеспечения, как средства реализации мета-эвристической оболочки (МЭО). Показано, что для рассматриваемых стандартов необходим обобщающий инструментарий, который если не объединит их между собой, то будет выступать отдельным инструментом, агрегирующим их возможности.

Общая постановка проблемы

Среди концепций инструментальных оболочек построения экспертных систем именно концепция мета-оболочки в наибольшей степени соответствует специфике САПР [1, 2]. Методика создания экспертной системы с помощью мета-оболочки предполагает построение концептуальной модели предметной области, определяемой как система взаимосвязанных уровней знаний о действительности (метазнаний).

К достоинствам базовой концепции мета-эвристических оболочек (МЭО) с точки зрения САПР можно отнести унифицированный подход, учитывающий специфику уровня и формы представления моделей в интеллектуальных САПР, наличие средств и методов пространственно-временного представления моделей и многие другие аспекты [1, 2, 3].

Специфика МЭО предполагает:

– описание моделируемого объекта в рамках пространственно-временной логики;

– представление модели структур на основе построения отношений достижимости между объектами, расположенными на различных созависимых абстрактных уровнях (уровнями декомпозиции, разделяющими систему на функциональные подсистемы);

– наличием внешних и внутренних границ функциональных блоков;

– физической семантикой, т.е. наличием инструментария свойственного объектно-ориентированным языкам программирования (ООП), с применением механизма «интеллектуальных шаблонов» (И-ИЛИ деревьев в ООП) [4].

Таким образом, предлагаемая концепция может рассматриваться как инструмент построения редактора онтологий, отличающуюся от существующих ориентацией на физическую семантику предметной области.

К недостаткам концепции МЭО можно отнести:

– относительно слабое исследование объектно-ориентированного подхода (ООП) как инструмента представления структур и функций в рамках иерархии моделей;

– отсутствие инструментальных средств работы с ООП (диаграммы типа UML).

Целью данной работы является:

– обсуждение принципов создания комплексного подхода, лишённого недостатков каждой из перечисленных концепций, но агрегируя всё положительное, чем они обладают;

– проведение анализа технологий структурного анализа, реализованных в диаграммах IDEF.

Современные тенденции в области использования нотаций

Современные тенденции в области нотаций направлены на замыкание всего цикла проектирования технологического процесса в едином комплексе диаграмм. Подобными тенденциями, широко используемыми на текущий момент, являются UML, IDEF, BPMN и прочими нотациями, внедрёнными в программную систему «Бизнес-инженер», а также SAP R/3.

Среди систем автоматизированного управления SAP R/3 вызывает сегодня, пожалуй, наибольший интерес (рис. 1). Наличие более тысяч инсталляций в мире делает систему R/3 одной из самых распространенных. Технологические особенности, заложенные в систему изначально, а также достижения последнего времени вывели её в число лидеров среди интегрированных систем управления.

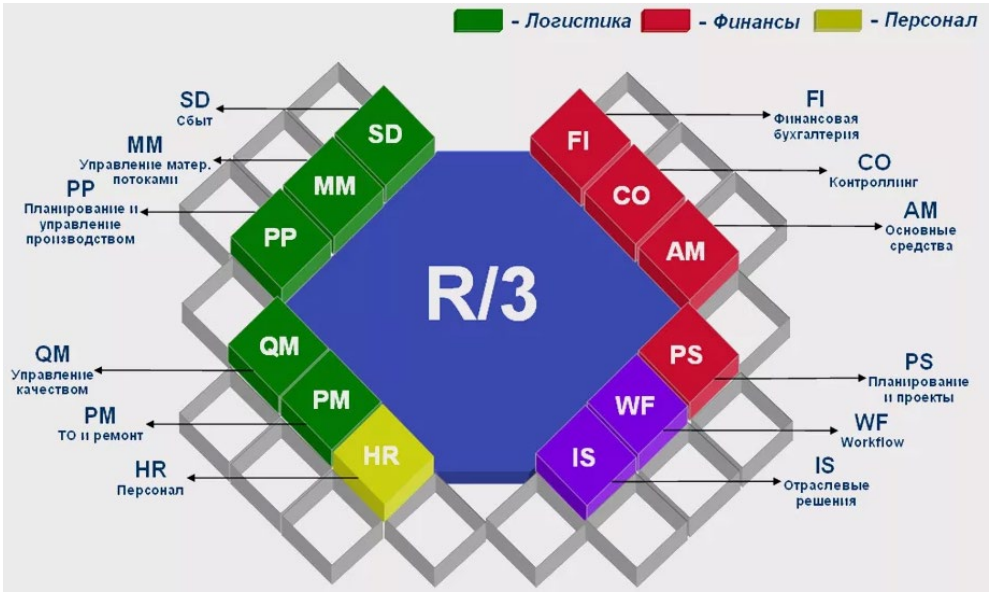


Рисунок 1 - SAP R/3 System Business Modules

Достоинства и недостатки IDEF нотаций

Диаграммы типа SADT/IDEF5 ориентированы на создание онтологий с точки зрения возможностей технологии SADT/IDEF0.

Диаграммы типа SADT/IDEF4 ориентированы на создание инструмента автоматизации разработки программы на основе ООП, т.е. их можно отнести к разряду CASE-технологий.

Недостатки данных систем:

– невозможность применения средств ООП SADT/IDEF4 в онтологии SADT/IDEF5;

– невозможность ориентироваться на физическую семантику предметной области как того требует задача построения интеллектуальных САПР.

Диаграммы типа SADT/IDEF0 ориентированы на создание блочных структур, которыми могут являться статические модели предметных областей и обладают рядом достоинств [5]:

– эффективная технология декомпозиции, которая позволяет рассматривать каждый блок, подвергаемый декомпозиции как набор внутренней границы и совокупность внутренних подблоков, связанных структурными связями;

– использование принципа «не более 7 подблоков»;

– автоматизация построения стрелочек, соединяющих те или иные блоки;

– классификация входов/выходов на классы, куда входят ресурсы, управляющие входы, информационные входы и информационные выходы;

– наличие специальной команды, позволяющей декомпозировать данный блок на требуемое число подблоков с указанием типа диаграмм (SADT/IDEF0, SADT/IDEF3), которые необходимо использовать для описания данной структуры.

Недостатками диаграмм типа SADT/IDEF0 являются:

– невозможность декомпозировать стрелочку, как тип данных, что не позволяет вводить абстрактные, а только структурные уровни;

– отсутствие явно прописанных составов входов/выходов каждого блока с указанием типов данных для данных входов/выходов;

– отсутствие списка связей, связывающих блоки между собой;

– отсутствие принадлежности блока к некоторому типу блоков и соответственно разделение имени на собственное имя блока и идентификатор типа блока;

– отсутствие наличия массивов блоков того или иного типа, а также размерности этого массива блоков.

Преодоление этих недостатков позволит:

– декомпозировать не только блоки, но и данные, относящиеся к тому и или иному типу (функция декомпозиция типов данных);

– декомпозировать стрелочки, т.е. структурное отношение передачи данных некоторого типа в случае декомпозиции типа данных (декомпозиция типа данных автоматически ведёт к декомпозиции всех стрелочек, соединяющие входы/выходы блоков по данному декомпозируемого типа);

– декомпозируется тип блока, что приводит к автоматической декомпозиции всех блоков данного типа, используемых во всех структурах – диаграммах, введённых ранее.

Кроме того, данный механизм декомпозиции позволит перейти к представлению и декомпозиции уже структурных модулей знаний, построенных как единичные онтологии.

Анализ возможности применения специфики диаграмм IDEF0 в рамках реализации МЭО

В области разработки средств структурной декомпозиции моделей в рамках технологий типа SADT/IDEF:

– поддержка технологий типа SADT/IDEF0 при декомпозиции структуры блоков с физической семантикой в случае декомпозиции по «И»;

– построения модуля знаний на основе использования набора системообразующих компонентов (блоков, структурных связей, внешних полюсов блоков типа входы/выходы);

– использование набора факультативных компонент как «И-ИЛИ» дерева;

– набора семантических связей, определяющих совместное одновременное существование альтернатив «ИЛИ» узлов посредством набора продукционных зависимостей.

На текущий момент имеется 15 стандартов IDEF, охватывающих практически все возможные функциональные данные: онтологии, ООП, потоки сетевых данных, интерфейсы программных систем и другие. Вот только в достаточно мере описаны лишь первые 6 стандартов, которые не лишены недостатков, в той или иной степени не позволяющих в должной мере использовать данный тип нотации при проектировании программных систем комплексно.

Модель IDEF0 – это графическое описание системы, созданное с определённой целью и с выбранной точки зрения, содержащее одну или больше диаграмм, которые изображают функции системы с помощью графики, текста и глоссария [6].

Декомпозиция является основной идеей структурного подхода, т.е. разделение системы на функциональные подсистемы. Декомпозиция системы до атомарного уровня (элементарных операций) представляет собой переход к иному уровню абстракции, т. е. выделению существенных и отвлечению от несущественных аспектов системы (рис. 2).

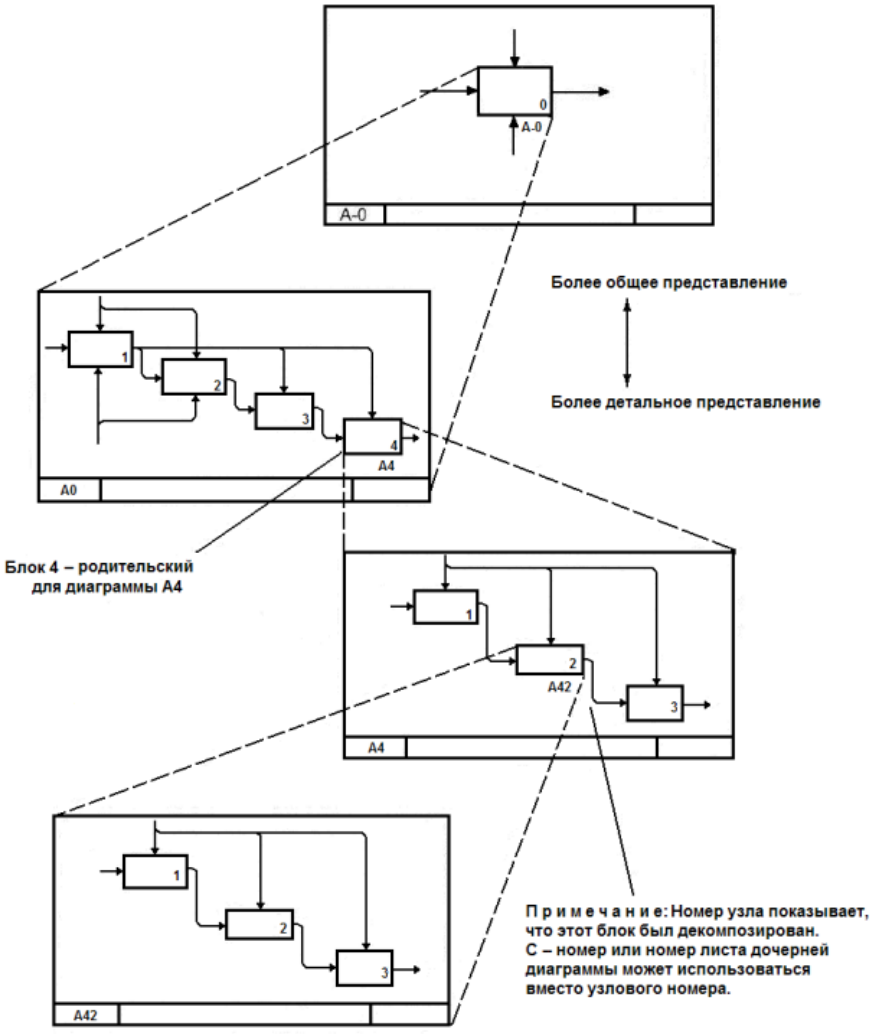


Рисунок 2 - Пример механизма декомпозиции диаграммы IDEF0

На верхнем уровне система представляет собой контекстную диаграмму, в которой на вход поступают материалы, а на выходе получают готовое изделие или брак. В качестве документа управления выступает маршрутная карта технологического процесса, в качестве механизмов – оборудование и оснастка, а также персонал участка сборки.

Из приведённого примера видно явное отличие от инструментальных средств UML, а именно: отсутствие инструментов позволяющих описывать и просматривать структуру классов (объектов, интерфейсов и прочих), взаимодействие и последовательность операций связанных непосредственно с каждым из объектов на каждом из уровней, что не полностью реализует идею абстрактных уровней с вероятной модификацией структурных связей объектов и отношений между ними.

Данная декомпозиция возможна в IDEF0, но только ручным способом, что значительно увеличивает объём структурно-функциональной модели. Каждый структурный блок должен раскладываться на необходимые для проектировщика схемы, диаграммы классов, правила и закономерности поведения объектов на каждом из уровней декомпозиции.

Модель IDEF0 также имеет существенный недостаток для оценивания системы в целом, который заключается в отсутствии возможности построить пространство поиска решений, соответствующих данному техническому заданию, что присуще онтологическому моделированию.

В перспективе развития принципов IDEF0, как средства реализации МЭО, акцентируем важность наличия инструмента, объединяющего декомпозированные диаграммы друг с другом, средства схожие с диаграммами классов и объектов, а также наличия механизма автоматического поиска решения, при условии статической модели представления данных.

Анализ возможности совмещения стандартов IDEF и объектно-ориентированного подхода как инструмента реализации МЭО

IDEF5 позволяет разрабатывать, изучать и поддерживать онтологию моделируемой системы.

Термин «онтология» включает в себя каталог терминов области знаний; правила, объясняющие, как термины могут комбинироваться, создавая при этом корректные ситуации в области знаний и согласованные выводы, используемые в моделируемой системе [7].

В рамках МЭО каждому классу в рамках диаграммы IDEF5 должен назначаться класс, описанный средствами IDEF4/C++ или физической семантикой, реализованной средствами используемого фреймворка или программного редактора, по типу аналога редактора онтологий.

Далее по описанной диаграмме должна быть возможность генерировать программные объекты (индивидуалы, но с переменными, а не включёнными данными) с выполнением, как минимум, примитивных арифметических операций.

Также программная система должна учитывать изменение состояния объекта в случае, если учитывает временную логику.

Специфика подобного описания реализована в стандарте IDEF4, где каждый объект представлен в 4х моделях: физический объект (как он есть в природе), объект-роль, событие-объект, взаимодействие, программный объект.

Каждая такая модель, по сути, представлена фреймом и должна быть описана средствами диаграммы IDEF3 (см. рис. 3), отражающей процесс взаимодействия между компонентами системы и структурную связь.

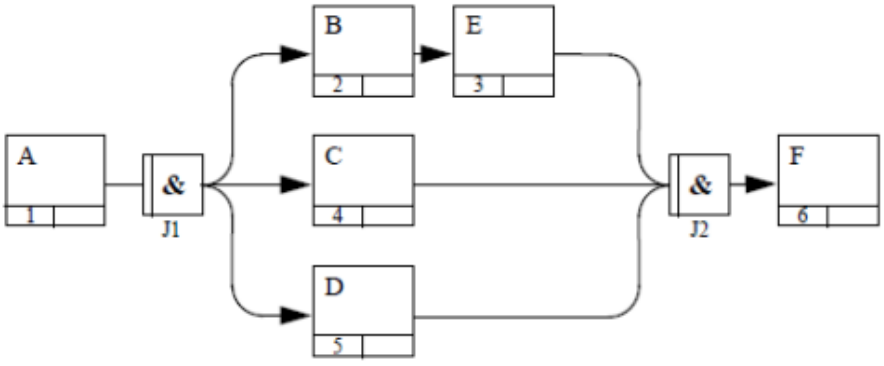


Рисунок 3 - Пример диаграммы IDEF3

Получаемая модель с физической семантикой также должна иметь инструмент декомпозиции любого из функциональных блоков сложной модели (см. рис. 2). Он должен позволять один и тот же процесс рассматривать с разных точек зрения, ролей сотрудников или технологического процесса, с возможностью генерировать программный код из формального описания.

Проблематика практического применения предлагаемого типа совмещённой диаграммы

Получаемый тип диаграммы позволяет в довольно сложной форме описывать множество простых технологических процессов и, в частности, замкнуть на себе полный цикл описания программного продукта, но является не читаемым для любых сложных проектов. Для примера, можно рассмотреть отдельно IDEF0, который в виду строгости описания технологических процессов за довольно короткое время обретает не читаемый вид и требует дополнительного описания выделенных аспектов с помощью иных нотаций [8].

В случае представления сложного комплексного технологического процесса читаемость подобной диаграммы будет являться рутинной задачей, что довольно быстро будет отторгнуто обществом и не получит должного развития в будущем. Этот недостаток потребует дополнительного этапа упрощения для комфортного чтения специалистами компании, либо заимствования описанных механизмов в иные инструменты описания технологических процессов.

Данная проблема должна быть решена созданием соответствующей интегральной технологии, объединяющей различные типы диаграмм в единое целое, посредством создания общего набора инструментов, обеспечивающих полноту набора средств для описания предлагаемого типа диаграмм.

Выводы

В данной работе обсуждены принципы создания комплексного подхода, лишённого недостатков каждой из перечисленных концепций, но агрегируя всё положительное, чем они обладают. Также проведён анализ технологий структурного анализа, реализованных в диаграммах IDEF.

Также проведён анализ возможности совмещения стандартов IDEF и объектно-ориентированного подхода как инструмента реализации мета-эвристической оболочки, с целью освещения проблематики практического применения предлагаемого типа диаграммы и появления в будущем более комплексного инструмента описания технологических процессов и программных продуктов.

В целом, набор стандартов IDEF в полной мере позволяет замкнуть полный цикл описания программного продукта, но в рамках средств МЭО этого недостаточно.

Для дальнейшего развития предложенной технологии следует рассмотреть инструменты популярных диаграмм UML, BPMN и других, с целью упрощения процесса чтения получаемых научных нотаций. Особенно если учитывать, что комплекс объединённых диаграмм будет на себе полностью замыкать весь процесс описания любого технологического процесса, имеющегося на текущий день, то жизненно необходимо снижать рутинный процесс проектирования и чтения получаемых нотаций.

Литература

1. Григорьев, А. В. Семантика модели предметной области для интеллектуальных САПР / А. В. Григорьев // Научные труды Донецкого государственного университета. Серия "Информатика, кибернетика и вычислительная техника", (ИКВТ-2000). – Донецк: ДонГТУ, 2000. – Вып. 10. - С. 148-154.

2. Григорьев, А. В. Унифицированная концептуальная модель предметной области / А. В. Григорьев // В кн. Информатика, кибернетика и вычислительная техника (ИКВТ-97). Сборник трудов ДонГТУ. - Донецк: ДонГТУ, 1997. - Вып. 1. - С. 225-228.

3. Григорьев, А. В. Особенности реализации мета-эвристической оболочки для построения САПР / А. В. Григорьев, А. А. Каспаров, Е. Н. Горшкова // Научные труды Донецкого государственного технического университета Серия: Проблемы моделирования и автоматизации проектирования динамических систем. – Донецк: ДонГТУ, 1999. – Вып. 10. - С. 217-222.

4. Григорьев, А. В. Анализ эффективности и перспектив развития методов построения двухсторонних трансляторов в задаче создания интеллектуальных надстроек над проблемно-ориентированными САПР / А. В. Григорьев, О. В. Морозова // Сборник научных трудов донецкого национального технического университета. Серия: «Вычислительная техника и автоматизация». – Донецк: ДонНТУ, 2011. – Вы. 20(182). - С.118-129.

5. Ovidiu S. Noran, Business modelling: UML vs IDEF, School of computing and information technology // Griffith university, 2011.

6. РД IDEF0-2000. Методология функционального моделирования. – М.: Издательство стандартов, 2000. – 75 с.

7. Палагин, А. В. Онтологические методы и средства обработки предметных знаний / А. В. Палагин, С. Л. Крывый, Н. Г. Петренко. –

[монография] – Луганск: изд-во ВНУ им. В. Даля, 2012. – 323 с.

8. Цуканова, О. А. Методология и инструментарий моделирования бизнес-процессов: учебное пособие. - СПб.: Университет ИТМО, 2015. – 100 с.

9. Инфостарт журнал. Новости индустрии автоматизации учета infostart.ru [Электронный ресурс] / Интернет-ресурс. - Режим доступа: https://infostart.ru/pm/1430187/ - Загл. с экрана.

10. Филипишин, Д. А. Анализ применения редакторов онтологий с физической семантикой в педагогической деятельности вуза / Д. А. Филипишин, А. В. Григорьев, Е. И. Приходченко Е.И. // Информатика и кибернетика. - Донецк: ДонНТУ, 2023. - № 2(32). – С. 47-52.

Филипишин Д.А., Григорьев А.В. Принципы IDEF как средства реализации мета-эвристической оболочки. *С целью повышения функциональности существующих концепций нотации информационных систем, которые не позволяют на текущий момент замкнуть весь цикл проектирования программных систем на себе выполнен анализ существующих стандартов нотации IDEF, как наиболее полно охватывающий все виды проектирования программного обеспечения, как средства реализации мета-эвристической оболочки (МЭО). Показано, что для рассматриваемых стандартов необходим обобщающий инструментарий, который если не объединит их между собой, то будет выступать отдельным инструментом, агрегирующим их возможности.*

Ключевые слова*: информационная система, нотации, мета-эвристическая оболочка, инструментарий.*

Filipishin D.A., Grigoriev A.V. Review of IDEF principles as a means of implementing meta-heuristic shell. *In order to increase the functionality of existing concepts of information systems notation, which currently do not allow closing the entire cycle of designing software systems on themselves, an analysis of existing IDEF notation standards was carried out, as it most fully covers all types of software design, as a means of implementing a meta-heuristic shell. It is shown that the standards under consideration require a generalizing toolkit, which, if it does not combine them with each other, will act as a separate tool that aggregates their capabilities.*

Keywords*: information system, notation, meta-heuristic shell, toolkit.*

Статья поступила в редакцию 14.04.2024
Рекомендована к публикации профессором Зори С.А.

УДК 681.5:629.056.6

Прецизионный инерциальный датчик угла наклона каната с фильтром Калмана

А. Ю. Грицаенко
Республиканский академический научно-исследовательский и проектно-конструкторский институт горной геологии, геомеханики, геофизики и маркшейдерского дела.
Донецкий национальный технический университет,
кафедра строительства зданий, подземных сооружений и геомеханики.
E-mail: anthony.yuriev@ya.ru

Аннотация
Выполнен анализ характеристик инерциальных датчиков, изготовленных по технологии микроэлектромеханических систем (МЭМС). Обоснована пригодность недорогих акселерометров и гироскопов для использования в автоматизированных системах обеспечения безопасной эксплуатации шахтных подъёмных установок. Разработан математический аппарат и встроенное ПО, позволяющие в реальном времени с высокой точностью определять угол наклона в сложных динамических системах, таких как «подъёмный канат - сосуд - армировка шахтного ствола». Описан опыт эксплуатации разработанных датчиков наклона каната на шахтах Донбасса.

Введение

Наиболее часто аварийные ситуации в шахтном стволе происходят при зависании подъёмного сосуда (скип, клеть). Так, при застревании сосуда во время движения вниз и дальнейшем вращении канатоведущего шкива происходит напуск каната на сосуд. Из-за большой массы каната, в некоторый момент происходит срывание сосуда, что зачастую приводит к обрыву каната или его деформации, а также к разрушению армировки ствола. Это вызывает не только простой оборудования во время ремонта, но и создаёт опасность гибели людей. Например, подобная авария произошла в 2012 году на шахте Южнодонбасская №1, где при падении в ствол скипа была разрушена армировка. Именно поэтому необходим непрерывный контроль напуска каната при движении сосуда в стволе, чтобы в момент его образования произвести своевременную остановку подъемной машины.

Существует несколько способов контроля напуска каната: контроль по верхней части или контроль по всей длине [1]. В первом случае контролируется провисание струны каната (для подъёмных установок барабанного типа), а во втором – локальное провисание каната непосредственно над сосудом, т.е. измерение натяжения или угла наклона каната в пространстве.

Последний метод универсален и пригоден для подъёмных установок любого типа: как барабанных, так и многоканатных со шкивами трения. Датчики наклона каната крепятся непосредственно на канате выше подвесного устройства, а передача информации на поверхность в систему управления подъёмной машины осуществляется посредствам радиоаппаратуры специального комплекта стволовой сигнализации и связи.

Цель данной работы – разработка математической модели инерциального датчика угла наклона каната, пригодного для эксплуатации в сложных условиях шахтного подъёма, а также его аппаратная и программная реализация на основе недорогого микроконтроллера.

Технология МЭМС и измерение угла наклона

Новой тенденцией в современной электронике является микросистемная техника, благодаря которой стало возможным создание миниатюрных инерциальных датчиков. МЭМС представляют собой совокупность классических электронных и механических элементов в микроисполнении.

Механические чувствительные элементы, построенные на вибрационных, ёмкостных, волновых или оптических принципах, преобразуют какое-либо воздействие на всю систему или отдельную её часть в электрический сигнал, который обрабатывает интегрированная электроника [2].

В недорогих МЭМС акселерометрах и гироскопах типовым элементом является балка и структуры на её основе в виде решёток монокристалла на подложке полупроводника. В общем случае это распределённая масса, позволяющая получать необходимый информационный сигнал емкостным, резонансным и другими методами (рис. 1).

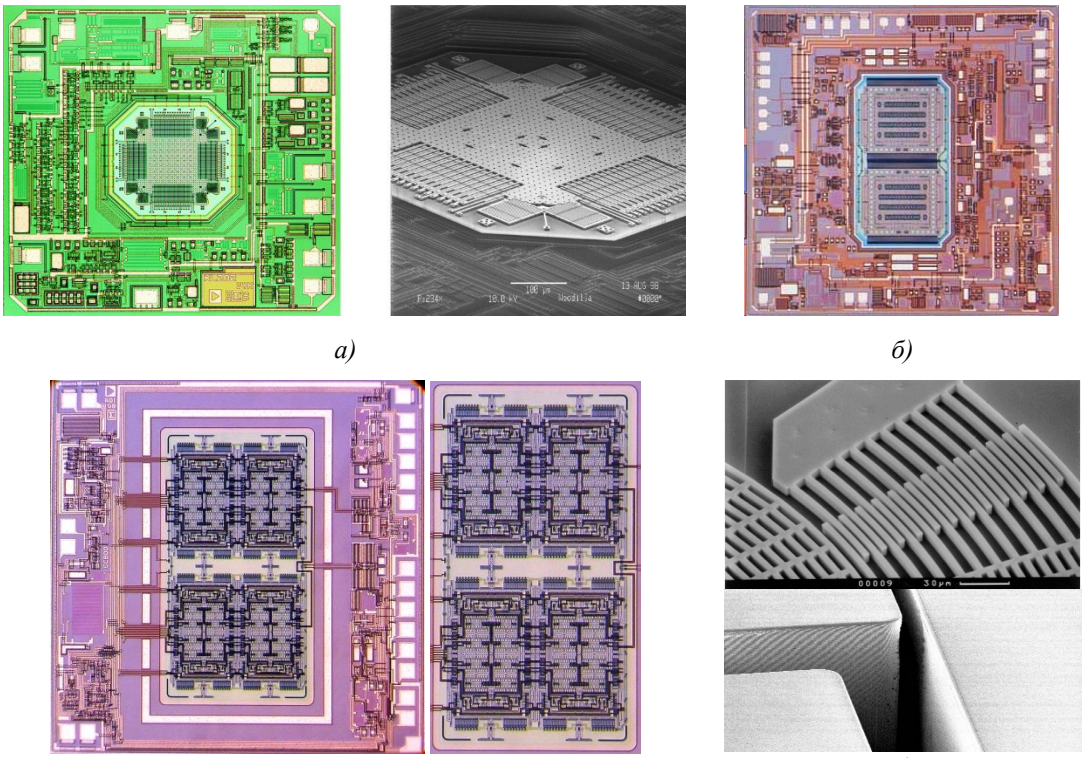


Рисунок 1 - Микроснимки МЭМС акселерометра ADXL203 (а), МЭМС гироскопов ADIS16100 (б) и ADXRS64x с разделёнными массами (в), масштабы типовых конструкции серийных вибрационных гироскопов (г)

Сигналы на выходе МЭМС акселерометра, находящегося в статическом состоянии, представляет собой проекции вектора силы тяжести на оси прибора. Но при движении подъёмного сосуда по рельсовым направляющим в стволе почти невозможно вычислить угол наклона каната лишь по данным акселерометра. Ввиду сложного характера динамики такого движения, в выходных сигналах датчика кроме реального и кажущегося ускорений всегда присутствует реакция на стохастические вибрации, резонансы и нестационарные колебания каната во всех плоскостях.

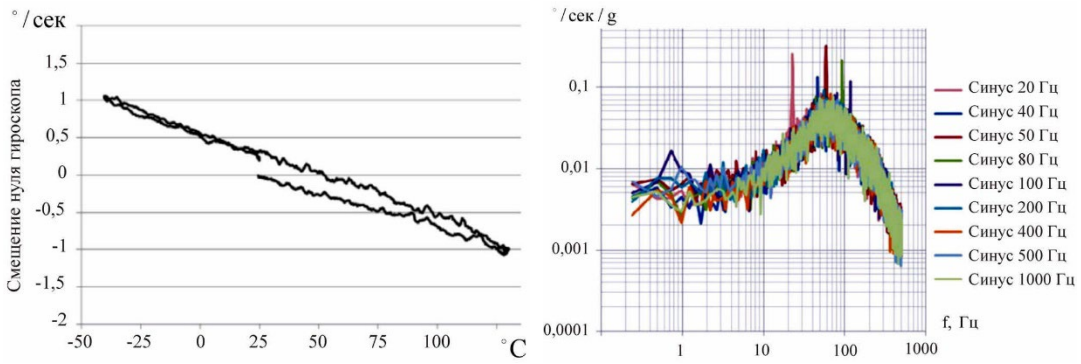
Поэтому для измерения угла наклона каната в движении помимо акселерометра нужно использовать не чувствительный к линейным ускорениям многоосевой МЭМС гироскоп, следуя принципам инерциальной навигации [3].

При этом главными источниками ошибок и шумов в данных МЭМС датчиков являются: ошибки смещений (постоянное смещение ноля, постоянная асимметрия ноля, случайная нестабильность ноля на длинной выборке), температурная нестабильность, случайное блуждание угла (гироскопы) и случайное блуждание скорости (акселерометры), ошибки квантования (для всех датчиков с цифровым выходом), дрейф угловой скорости (гироскопы) и чувствительность к синусоидальной вибрации.

Исходя из данных экспериментальных исследований, систематический дрейф МЭМС гироскопов, пригодных для решения задач контроля напуска каната, не должен превышать 0,5-2°/час. В свою очередь подобный «уход» и другие параметры неидеального гироскопа компенсируются с помощью акселерометра. Важны также крутизна характеристики мВ/°/сек, полоса пропускания гироскопа Гц, температурный дрейф °/°С/сек и отношение сигнал/шум на выходе датчика. Способность к подавлению синусоидальных вибраций МЭМС гироскопов с физической точки зрения зависит от динамики самого чувствительного элемента. Например, на рис. 2, а показан циклический уход нуля не термостатированного гироскопа ADXRS453.

На рис. 2, б – реакция гироскопа серии CRG20 на вибрации в широком диапазоне частот. Учитывая сложные условия эксплуатации подъёмных установок и электронного оборудования из состава автоматизированных систем обеспечения безопасности, находящегося на подъёмном сосуде, перечисленные параметры являются основными при выборе гироскопов для построения датчика угла наклона каната.

В свою очередь к акселерометрам предъявляются не столь строгие требования. Так, рабочий диапазон линейных ускорений МЭМС акселерометров может быть в интервале 1-2g, а возможные перегрузки при больших ускорениях компенсируются выносливостью прибора.



а) *б)*

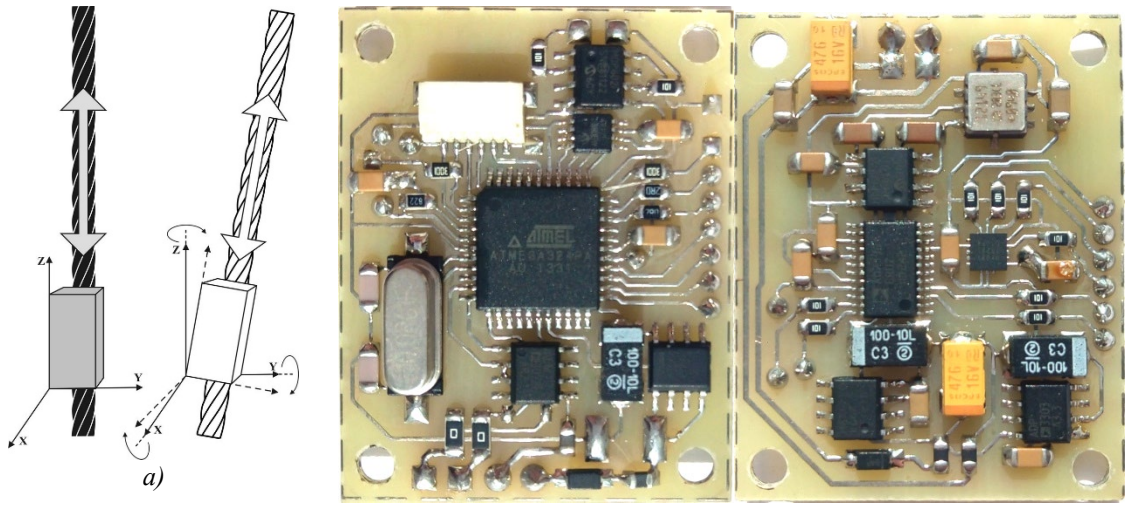
Рисунок 2 - Кривая температурного гистерезиса гироскопа ADXRS453 при некомпенсированном смещении нуля (а), реакция гироскопа CRG20 на асимметричные осям прибора вибрации (б)

Как правило, для МЭМС приборов с сосредоточенными параметрами допустимы перегрузки в 10000-30000g, т.е. они заведомо выдерживают падения и удары.

Устройство инерциального датчика наклона каната

Конструктивно датчик наклона каната выполняется в ударопрочном стальном корпусе. Крепления в виде хомутов обеспечивают его надёжную и долговременную фиксацию на канате. Внутри корпуса расположены электронные платы на резиновых демпферах для уменьшения вибраций в широком спектре частот. На плате располагаются микроконтроллер с необходимыми вычислительными возможностями, гироскоп и акселерометр, ориентированные специальным образом, усилители и АЦП, трансиверы цифровых и аналоговых интерфейсов.

Инерциальная часть датчика состоит из трёхосных гироскопа и акселерометра. Алгоритм определения угла наклона (АУН) строится на базе измерений акселерометра, измеряющего проекции кажущегося и реального ускорения на свои оси чувствительности, и гироскопа, измеряющего абсолютную угловую скорость. В основе АУН лежит интегрирование ускорений и угловых скоростей. В данном случае объект измерений – движущийся канат, а данные с датчиков интегрируются в проекциях на выбранную систему координат. Для текущей задачи подходит ортогональная локальная система координат с направленной по канату одной из осей (рис. 3а). На рис. 3б показан один из вариантов конструктивного исполнения платы датчиков наклона каната, разработанных автором и выпускающихся серийно. Датчики имеют точность 0,01° в статике и 0,03° в динамике.



а) *б)*

Рисунок 3 - Датчик угла наклона на канате в привязке к локальной системе координат (а) и конструктивное исполнение второго поколения платы датчика наклона каната на основе акселерометра ADXL203, гироскопа MAX21000 и микроконтроллера ATmega324PA (б)

Оси чувствительности акселерометра образуют с канатом связанную систему координат. Выходной сигнал по всем трём осям является проекцией кажущегося ускорения на оси локальной системы координат. Основная сложность в том, что измерения акселерометра одновременно содержат информацию об абсолютном и гравитационном ускорениях. Абсолютное ускорение содержит информацию о движении каната относительно земли и его собственном вращательном движении.

Таким образом, измерения акселерометра представляют собой аналоговый сигнал, нуждающийся в компенсации ошибок. В этой части модели компенсируется смещение нуля акселерометра, погрешность установки, а также нестабильность масштабного коэффициента вследствие воздействия значительных линейных ускорений при движении сосуда, крутильных и продольных колебаний каната.

Численная реализация АУН для микроконтроллера по дискретным значениям угловой скорости вибрационного гироскопа и углового перемещения акселерометра является задачей нетривиальной вследствие высокого уровня помех в широкой полосе частот (вибрации, погрешности первичных преобразователей, вызванные воздействием парциальных частот колебаний системы «подъёмный сосуд - канат») и шумов квантования АЦП. Инерциальные чувствительные элементы выдают информацию о приращении углов и угловых скоростей с частотой от 100 до 1000 Гц, чего достаточно для контроля аварийного напуска каната с быстродействием 10-50 миллисекунд.

Для выделения полезного сигнала МЭМС датчиков из аддитивной помехи (основой которой являются собственные частоты колебаний каната, удары и вибрации при движении сосуда) используется рекурсивная фильтрация [4,5]. Одним из подходов к данной задаче также является калмановская фильтрация. Имея модель системы и экспериментально определённые статистические свойства сигналов и помехи, можно построить оптимальную передаточную функцию фильтра [6], минимизировав среднеквадратичную ошибку определения угла. Для такого АУН не требуется больших вычислительных мощностей, достаточно микроконтроллера. А вычисление состояния системы в режиме реального времени – это обязательное требование для реализации аварийных защит. При этом неточно выбранные параметры математической модели компенсируются гибкостью алгоритма.

В общем случае модель данной линейной системы может быть представлена в пространстве состояний двумя уравнениями:

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k + w_k, \\ y_k &= C_k x_k + z_k. \end{aligned} \qquad (1)$$

где A_k, B_k, C_k – матрицы, причём A_k – квадратная;

x_k – вектор переменных состояния системы, который является случайным Гауссовским процессом;

u_k – известный вектор входных переменных;

y_k – измерения, полученные в момент времени t_k (выход системы);

w_k, z_k – шумы соответственно моделируемого процесса и результатов измерений.

Вектор x_k содержит всю информацию о текущем состоянии динамической системы, но не поддаётся непосредственному измерению. Измеряется вектор y_k, каждая составляющая которого в общем случае является функцией вектора x_k и шумов измерений z_k. Поэтому можно использовать y_k для оценки вектора состояния системы x_k. Шумы системы и измерений w_k и z_k также являются Гауссовскими случайными процессами с нулевым математическим ожиданием (белый шум). Задачей фильтрации является нахождение оценки вектора состояния системы x_k, являющейся функцией измерений y_k, которая минимизировала бы среднеквадратичную ошибку.

Фильтр работает по принципу «оценка – коррекция». Пусть в момент времени t_k получена оценка вектора состояния системы x_k, а нужно получить оценку в точке t_{k+1}. Строится прогноз оценки $\hat{x}_{k+1}$, базируясь на x_k, получаются измерения y_k с последующей коррекцией оценки состояния системы $\hat{\hat{x}}_{k+1}$ в момент t_{k+1} по прогнозу и измерениям. Так получается окончательная оценка вектора состояния $\hat{\hat{x}}_{k+1}$ в момент времени t_{k+1}. Принято $\hat{x}_{k+1}$ называть априорной оценкой, а $\hat{\hat{x}}_{k+1}$ – апостериорной.

В случае моделирования движения датчика наклона и подъёмного каната вектор

состояния системы «датчик – канат» в простейшем случае можно сократить до двух компонент: текущего угла наклона и угловой скорости (корпуса датчика относительно ортогональной системы координат, связанной с канатом). Вход системы ω_k – угловая скорость, а выход – измеренный угол наклона θ_k. Предположим, что мы в состоянии измерять угловую скорость и угол наклона с периодом dt секунд. Тогда угол наклона на шаге t_{k+1} задаётся следующим уравнением:

$$\theta_{k+1} = \theta_k + \omega_k dt\,, \tag{2}$$

где ω_k – результат измерения угловой скорости в момент времени t_k.

Т.е. текущий угол равен углу на предыдущем этапе измерений плюс текущее значение угловой скорости, умноженное на длину временного интервала измерения. Но предыдущее уравнение не даёт точного значения угла наклона ввиду наличия упомянутых ранее случайных процессов и шумов, меняющихся со временем. Поэтому более реалистичное уравнение запишется так:

$$\theta_{k+1} = \theta_k + \omega_k dt + \overline{\theta}_k, \tag{3}$$

где $\overline{\theta}_k$ – шумы системы.

Учитывая, что вектор состояния системы $x_k = \begin{bmatrix} \theta_k \\ \omega_k \end{bmatrix}$, можно записать уравнения состояния системы (1) в матричной форме:

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} \theta_{k+1} \\ \omega_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & -dt \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_k \\ \omega_k \end{bmatrix} + \begin{bmatrix} dt \\ 0 \end{bmatrix} u_k + w_k, \\ y_{k+1} &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_k \\ \omega_k \end{bmatrix} + z_k. \end{aligned} \tag{4}$$

Полученный АУН должен удовлетворять двум свойствам: а) среднеквадратичное значение ошибки алгоритма должно «совпадать» с ошибкой самой системы. Т.е. математическое ожидание оценки состояния должно мало отличаться от ожидаемого состояния самой системы; б) алгоритм должен обладать минимальной собственной ошибкой или вариацией.

Именно фильтр Калмана является идеальным средством оценки, удовлетворяющим двум свойствам выше. Но для использования калмановских алгоритмов нужно наложить некоторые ограничения на характер относительно шума, воздействующего на систему. Из нашей модели w_k – шум динамической системы «датчик - канат», z_k – шум процесса измерений. Предположим, что нормальное значение $w_k = 0,\ z_k = 0$. Более того, не должно существовать корреляции между w_k и z_k (т.е. это независимые случайные переменные в любой момент t_k). Второй момент случайного процесса описывается в терминах ковариационной матрицы

$$\begin{aligned} S_w &= E\left(w_k w_k^T\right), \\ S_z &= E\left(z_k z_k^T\right), \end{aligned} \tag{5}$$

где S_w, S_z – ковариационные матрицы ошибки оценки вектора состояния;

z_k^T, w_k^T – транспонированные матрицы случайных шумов;

$E(\ )$ – ожидаемое среднеквадратичное значение ошибки.

Опуская интегрирование модельной динамической системы уравнений (1) для получения прогнозируемой оценки ковариационной матрицы ошибки, запишем непосредственно уравнения фильтра Калмана:

$$K_k = AP_kC^T\left(CP_kC^T + S_z\right)^{-1}, \tag{а}$$

$$\hat{x}_{k+1} = \left(A\hat{x}_k + Bu_k\right) + K_k\left(y_{k+1} - C\hat{x}_k\right), \tag{б}$$

$$P_{k+1} = AP_kA^T + S_w - AP_kC^TS_z^{-1}CP_kA^T, \tag{в}$$

(6)

Где K_k – коэффициент усиления фильтра (или матрица коэффициентов обратной связи);

P_k – ковариация ошибки прогнозирования.

В уравнении 6, а с ростом шумов измерений растёт ковариационная матрица S_z, а значит коэффициент усиления фильтра будет уменьшаться вместе с достоверностью оценки состояния системы. Т.е. эта матрица коэффициентов является функцией от априорного значения ковариационной матрицы ошибки. С уменьшением шумов S_z достоверность прогнозирования растёт.

Первое слагаемое в уравнении 6б повторяет модель линейной системы (1) при отсутствии средств измерения. Второе слагаемое

в 6б называется коррекционным членом и показывает, насколько необходимо скорректировать прогнозируемое значение интересующей величины в соответствии с текущим измерением (действительным измерением тут нужно считать показания акселерометра, содержащие данные о проекции на оси чувствительности гравитационного ускорения).

Структура программы микроконтроллера, построенной в соответствии с описанным алгоритмом, показана на рис. 4.

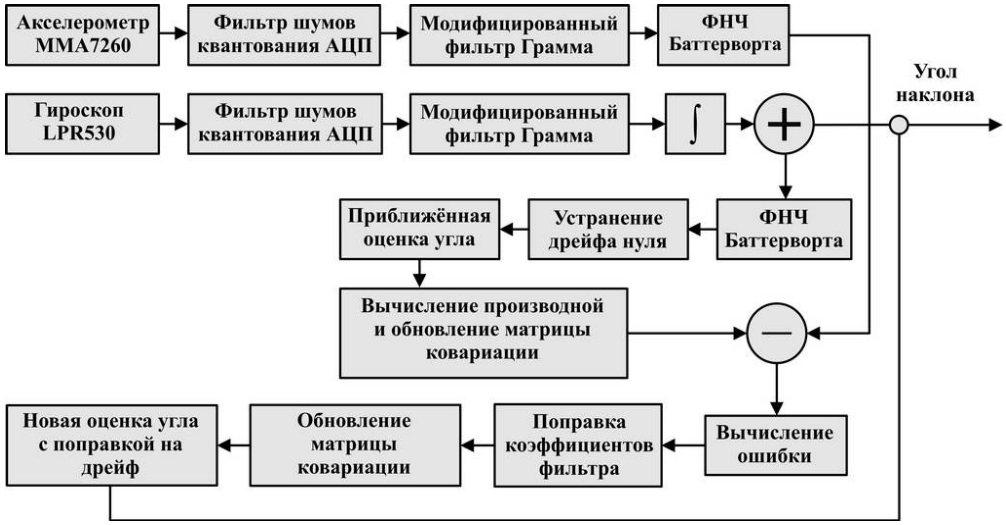


Рисунок 4 – Практическая реализация алгоритма вычисления угла наклона каната с использованием рекурсивного фильтра Калмана (в основе датчиков первого поколения)

Предварительная и промежуточная обработка сырых данных с МЭМС датчиков осуществляется посредствам модифицированных фильтров низких частот [7, 8] с минимальным или отрицательным запаздыванием.

Для примера рассмотрим реализованный в алгоритме АУН рекурсивный фильтр Баттерворта 4го порядка. Фильтр построен в виде последовательного соединения секций фильтров второго порядка для ФНЧ.

При этом выходной сигнал фильтра второго порядка (одной секции) записывается как:

$$y(t_n) = \left(a_0 x(t_n) + a_1 x(t_{n-1}) + a_2 x(t_{n-2})\right) - \left(b_1 y(t_{n-1}) + b_2 y(t_{n-2})\right), \quad (7)$$

где $a_0 a_1, a_2, b_1, b_2$ - коэффициенты фильтра, которые рассчитываются для конкретной частоты дискретизации и частот $f_{\min}$ или $f_{\max}$;

$x(t_n)$ - точка данных временного ряда в момент времени t_n;

$y(t_n)$ - посчитанное выходное значение секции.

Величина $y(t_n)$ является входной для следующей секции фильтра. При этом на каждой $n+1$ итерации алгоритма осуществляется формальная замена данных во временном ряду, так что $x(t_{n-2}) = x(t_{n-1})$, $x(t_{n-1}) = x(t_n)$, $x(t_n) = y(t_n)$. Коэффициент передачи каждой секции такого фильтра равен:

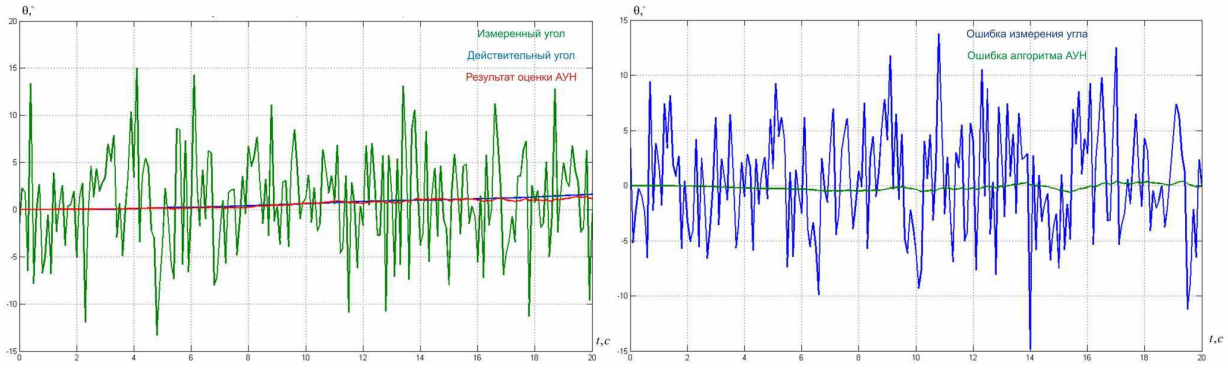
$$H(i\omega) = \frac{\left(a_0 + a_1 e^{(-i\omega)} + a_2 e^{(-2i\omega)}\right)}{\left(b_0 + b_1 e^{(-i\omega)} + b_2 e^{(-2i\omega)}\right)}, \quad (8)$$

где $\omega = \frac{2\pi f_0}{f_{дискр}}$ – угловая частота.

Общий коэффициент передачи вычисляется как произведение коэффициентов каждой секций. Для секций четвертого порядка индексы коэффициентов в формулах расширяются до четырех.

Моделирование и обсуждение результатов

На рис. 5 показаны результаты моделирования системы (4) с условиями Калмана (6) в среде Matlab для одной плоскости измерений.



а) *б)*

Рисунок 5 – Результаты моделирования АУН с фильтром Калмана в среде Matlab (а), сравнение абсолютной ошибки измерения угла с выходом фильтра (б) для одной плоскости измерений

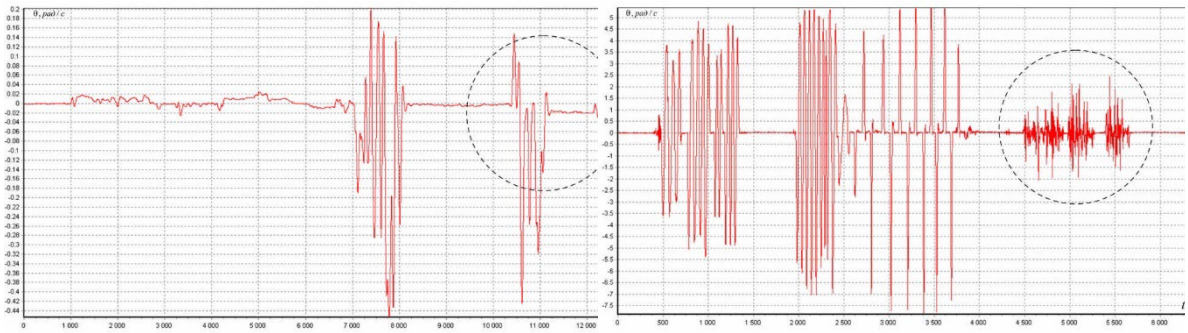
В качестве ковариаций S_z, S_w брались квадраты скалярных значений соответствующих шумов s_z, s_w. Сама линейная система симулировалась на каждом шаге с помощью генератора случайных чисел. Из рисунков видно, что соответствующим образом настроенный фильтр обладает необходимыми свойствами подавления случайных выбросов в показаниях первичных датчиков. Получение абсолютной величины отклонений каната от вертикали в трёх измерениях не составляет затруднений.

Ниже показаны результаты экспериментальных исследований для датчиков первого поколения на основе гироскопа LPR530 и акселерометра MMA7260. На рис. 6, а показана зависимость ухода ноля гироскопа от времени по одной из осей (сырые данные). На рис. 6, б – нормальная работа (после компенсации ухода ноля) и нежелательная реакция этого же гироскопа на линейные ускорения до $3g$ и вибрации в диапазоне частот до 1 кГц на испытательном стенде.

На рис. 7, а показаны данные с гироскопа LPR530 с подобранными коэффициентами фильтров АУН. Частота среза первичной секции фильтров здесь 100 Гц в соответствии с необходимой характеристикой быстродействия датчика. По сравнению с рис. 6 видно, что правильный подбор коэффициентов фильтров АУН полностью восстановил сигналы угловых скоростей с гироскопа несмотря на шумы в высокочастотной области.

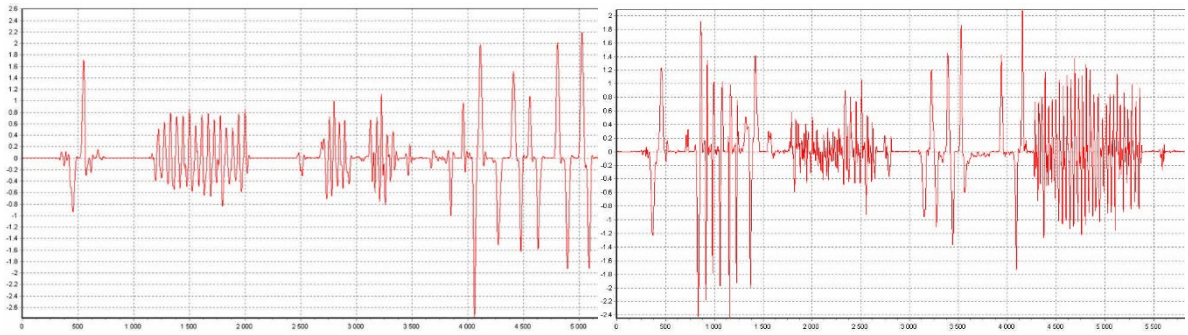
На рис. 8 приведены результаты измерения угла наклона каната при движении клети в восточном воздухоподающем стволе шахты Должанская-Капитальная.

Разработанный датчик входил в комплекс технических средств обеспечения безопасности подъёмных установок (КТСБПУ) производства НИИГМ имени М.М. Фёдорова. Передача данных с датчика велась посредствам аппаратуры стволовой связи СКРС-1Д. Программа датчика реализует описанный выше алгоритм АУН.



а) *б)*

Рисунок 6 – Результаты экспериментальных исследований гироскопа LPR530 в составе датчика наклона каната без компенсации алгоритма АУН (а), то же, но компенсацией дрейфа и воздействием вибраций (б)

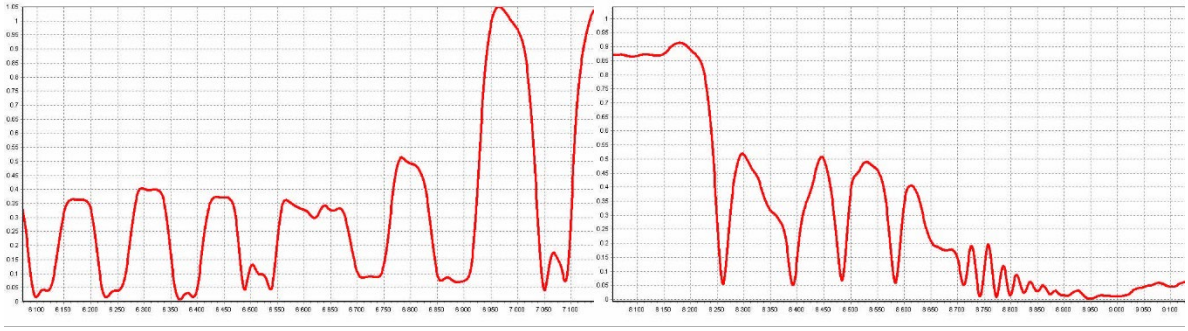


а) *б)*

Рисунок 7 – Данные гироскопа LPR530 после подбора коэффициентов фильтра. Без воздействия вибраций (а) и с вибрациями на канате в широком диапазоне частот (б)

На рис. 8, а показано естественное среднеквадратичное отклонение угла наклона каната от вертикали непосредственно над клетью в процессе движения клети вниз. Видно, что максимальная величина отклонения нормальной к канату плоскости X-Y от вертикали Z достигает 1 градуса.

На графике рис. 8, б показаны результаты измерения угла при посадке клети на жёсткое основание. На графиках явно присутствует колебательный процесс, связанный с естественной динамикой продольных и поперечных колебаний в системе подъёмный сосуд - канат.



а) *б)*

Рисунок 8 – Угол наклона каната, измеренный датчиком в процессе движения клети вниз (а) и после посадки её на жёсткое основание (б)

Также видно, что после посадки клети на жёсткое основание (в момент загрузки) колебания на канате затухают, что отражено на графике справа. Период этих колебаний также определяется динамикой и переходными характеристиками подъёмной установки.

При достижении углом наклона каната некоторого заданного порога (на практике это 3-5 градусов) датчик сигнализирует об аварии.

Связь с вышестоящими блоками автоматики организуется по интерфейсу RS485 или CAN в зависимости от модификации датчика.

Выводы

Эксплуатация разработанных автором датчиков наклона каната и их серийное производство ведётся НИИГМ имени М.М. Фёдорова с 2012 года. Опытные образцы датчиков наклона каната в составе КТСБПУ были испытаны на подъемной установке восточного воздухоподающего ствола шахты «Должанская-Капитальная» ГП «Свердловантрацит». Установочная серия поставлена на ОП «Шахта «Вергелевская» (4 комплекта) и ОП «Шахтоуправление «Луганское» (3 комплекта) ГП «Луганскуголь», ОП «Шахта «Белозерская» (3 комплекта) ГП «Добропольеуголь», ГП «Шахтоуправление «Южнодонбасское №1» (2 комплекта), ПАО «Краснодонуголь» (10 комплектов), ПАО «Павлоградуголь» (2 комплекта).

В составе комплексной системы автоматизации шахт УТАС разработки ГП «Машиностроительный завод «ИТРАС» (ранее «Петровский завод угольного машиностроения») данные датчики производились наряду с другим техническим оснащением подсистем КТСБПУ. Датчики показали свою надёжность, а разработанный алгоритм – устойчивость к широкому спектру возмущающих воздействий.

Литература

1. Шахтный подъем: научно-производственное издание / В. Р. Бежок, В. И. Дворников,

И. Г. Манец, В. А. Пристром; общ. Ред. Б. А. Грядущий, В. А. Корсун. – Донецк: ООО «Юго-Восток, Лтд», 2007. – 624 с.

2. Нанотехнологии в электронике. Монография. / Под ред. Ю. А. Чаплыгина. – М.: Техносфера, 2005. – 448 с.

3. Басараб, М. А. Математическое моделирование физических процессов в гироскопии. Монография / М. А. Басараб, В. Ф. Кравченко, В. А. Матвеев. – М.: Радиотехника, 2005. – 176 с.

4. Хэмминг, Р. В. Цифровые фильтры: пер. с англ. / Под ред. А. М. Трахмана. – М.: Советское радио, 1980. – 224 с.

5. Васильев, В. П. Основы теории и расчёта цифровых фильтров: учебное пособие для высших учебных заведений / В. П. Васильев, Э. Л. Муро, С. М. Смольский; под редакцией С. М. Смольского. – М.: Издательский центр "Академия", 2007. – 272 с.

6. Балакришнян, А. В. Теория фильтрации Калмана: перевод с английского – М.: Мир, 1988. – 168 с.

7. Айфичер, Э. Цифровая обработка сигналов, практический подход, 2-е издание. Пер. с англ. / Э. Айфичер, Б. Джервис. – М.: Издательский дом "Вильямс", 2004. – 992 с.

8. Гольдберг, Л. М. Цифровая обработка сигналов: Учебное пособие для вузов / Л. М. Гольдберг, Б. Д. Матюшкин, М. Н. Поляк. – 2-изд., перераб. и доп. – М.: Радио и связь, 1990. – 256 с.

Грицаенко А. Ю. Прецизионный датчик угла наклона каната с фильтром Калмана. *Выполнен анализ характеристик инерциальных датчиков, изготовленных по технологии микроэлектромеханических систем (МЭМС). Обоснована пригодность недорогих акселерометров и гироскопов для использования в автоматизированных системах обеспечения безопасной эксплуатации шахтных подъёмных установок. Разработан математический аппарат и встроенное ПО, позволяющие в реальном времени с высокой точностью определять угол наклона в сложных динамических системах, таких как «подъёмный канат - сосуд - армировка шахтного ствола». Описан опыт эксплуатации разработанных датчиков наклона каната на шахтах Донбасса.*

Ключевые слова: *акселерометр, гироскоп, микроконтроллер, фильтр низкой частоты, фильтр Калмана, угол наклона каната, шахтная подъёмная установка, вибрации.*

Gritsaenko A. Yu. Precision inclination angle sensor with Kalman filter for mine steel ropes. *The article carried out an analysis of characteristics for inertial sensors manufactured with microelectromechanical systems (MEMS) technology. Substantiated suitability of inexpensive accelerometers and gyroscopes for use as part of automated equipment ensuring the safe operation of mine hoist systems. Also, mathematical apparatus and built-in software have been developed that make it possible to determine in real time the angle of inclination in complex dynamic systems, such as “lifting rope – vessel – mine shaft reinforcement” with high accuracy. The experience of operating the developed rope tilt sensors in Donbass mines is described.*

Keywords: *accelerometer, gyroscope, microcontroller, low-pass filter, Kalman filter, steel rope angle, mine hoist, vibrations.*

Статья поступила в редакцию 15.04.2024
Рекомендована к публикации профессором Мальчевой Р.В.

Об авторах

Айдин Сергей Анатольевич - студент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Бездетный Николай Артемович - аспирант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Бирюков Алексей Борисович - доктор технических наук, профессор, проректор, заведующий кафедрой технической теплофизики факультета металлургии и теплоэнергетики ФГБОУ ВО «Донецкий национальный технический университет».

Боднар Алина Валериевна - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Зори Сергей Анатольевич – доктор технических наук, доцент, заведующий кафедрой программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Григорьев Александр Владимирович - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Гридин Сергей Васильевич - кандидат технических наук, доцент, доцент кафедры промышленной теплоэнергетики факультета металлургии и теплоэнергетики ФГБОУ ВО «Донецкий национальный технический университет».

Грицаенко Антон Юрьевич – научный сотрудник республиканского академического научно-исследовательского и проектно-конструкторского института горной геологии, геомеханики, геофизики и маркшейдерского дела; инженер кафедры строительства зданий, подземных сооружений и геомеханики ФГБОУ ВО «Донецкий национальный технический университет».

Мальчева Раиса Викторовна - кандидат технических наук, доцент, доцент кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Оверчук Иван Дмитриевич - магистрант кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Терещенко Кирилл Александрович - аспирант кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Филипишин Дмитрий Александрович - аспирант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Чередникова Ольга Юрьевна – кандидат технических наук, доцент, доцент кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

**Требования к статьям,
направляемым в редакцию научного журнала
«Информатика и кибернетика»**

Редколлегией принимаются к рассмотрению статьи, в которых рассматриваются важные вопросы в области информатики и кибернетики. Научный журнал издаётся с 2015 года, периодичность издания – 4 раза в год.

В журнале предусмотрены следующие рубрики:

- информатика и вычислительная техника;
- компьютерные и информационные науки;
- инженерное образование.

В соответствии с номенклатурой специальностей научных работников МОН ДНР первые две рубрики соответствуют следующим укрупненным группам специальностей научных работников:

05.01 – «Инженерная геометрия и компьютерная графика» (по новому перечню соответствует группе 1.2 «Компьютерные науки и информатика»);

05.13 – «Информатика, вычислительная техника и управление» (соответствует группе 2.3 «Информационные технологии и телекоммуникации»).

С 01.02.2019 Научный журнал включён в Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание учёной степени кандидата наук, на соискание учёной степени доктора наук (приказ МОН ДНР № 135) по группам специальностей 05.01.00 и 05.13.00.

Рубрика «Инженерное образование» предназначена опубликования сотрудниками научно-методических статей.

Журнал также включён в базу данных РИНЦ (Российский индекс научного цитирования) (лицензионный договор № 425-07/2016 от 14.07.2016).

Статьи, представляемые в данный сборник, должны отвечать следующим требованиям. ***Содержание статьи*** должно быть посвящено актуальным научным проблемам и включать следующие необходимые элементы:

- постановку проблемы в общем виде, её связь с важными научными и практическими задачами;
- анализ последних исследований и публикаций, в которых решается данная задача и на которые опирается автор, выделение нерешенных ранее частей общей проблемы, которым посвящается статья;
- формулировка цели статьи и постановка задач, решаемых в ней;
- изложение основного материала с полным обоснованием полученных научных результатов;
- выводы и перспективы последующих исследований в данном направлении.

Каждый элемент должен быть выделен соответствующим названием раздела, например, «введение», «постановка задачи», «цель и задачи работы», «цель статьи», «цель исследования», «цель разработки», «анализ … », «сравнительная оценка … », «разработка … », «проектирование … », «программная реализация», «тестирование … », «полученные результаты», «выводы», «литература». Разделы «введение», «выводы», «литература» являются обязательными. Включать в названия разделов нумерацию не разрешается.

В основном тексте статьи формулируются и обосновываются полученные авторами утверждения и результаты. Выводы должны полностью соответствовать содержанию основного текста. Языки публикаций: русский, английский.

Объём статьи, формат страницы

Для оформления статьи следует использовать листы формата А4 (210х297 мм) с полями по 2,5 см со всех сторон. Нумерацию страниц выполнять не нужно.

Рекомендуемый объём статьи – 6-12 страниц. Рукописи меньшего объёма могут быть рекомендованы к публикации в качестве коротких сообщений.

Последняя страница текста статьи должна быть заполнена не менее чем на две трети, но содержать не менее трёх пустых строк в конце.

Форматирование текста

Подготовка статьи осуществляется в текстовом редакторе Microsoft Office Word.

Весь текст статьи оформляется шрифтом Times New Roman 10 пт с одинарным междустрочным интервалом, если ниже в требованиях не сказано иного. Абзацный интервал «перед» – 0 пт, «после» – 0 пт.

На первой строке с выравниванием по левому краю располагается УДК.

Заголовок (название) статьи оформляется шрифтом Times New Roman 14 пт, полужирное начертание, с выравниванием по центру (без абзацных отступов). Заголовок статьи следует печатать с прописной буквы без точки в конце, переносы слов не допускаются. Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

После названия статьи следует информация об авторах, которая выравнивается по центру (без абзацных отступов). На одной строке указываются инициалы и фамилии всех авторов через запятую. Между двумя инициалами ставится пробел. С новой строки указывается название вуза (организации) и город (для каждого автора, если не совпадают). На следующей строке указываются адреса электронной почты (один адрес либо каждого автора – по желанию). Адрес электронной почты оформляется в виде гиперссылки.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Заголовок «Аннотация» выделяется полужирным начертанием. Объём аннотации – 450-550 символов (без пробелов). Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

Основной текст статьи разбивается на две колонки шириной по 7,5 см (промежуток между столбцами – 0,99 см), выравнивается по ширине. Абзацный отступ первой строки – 1 см. Автоматический перенос слов не применяется.

Заголовки разделов выполняются шрифтом Arial 10 пт, полужирное курсивное начертание. Абзацный отступ отсутствует, интервал перед абзацем – 12 пт, после абзаца – 6 пт. Для заголовка «Введение» установить интервал «перед» – 0 пт, «после» – 6 пт.

Таблицы в тексте статьи

Название следует помещать над таблицей с абзацного отступа (1 см) в формате: слово «Таблица», пробел, номер таблицы, пробел, тире, пробел, название таблицы. Название таблицы записывают с прописной буквы без точки в конце строки и выравнивают по ширине. В ячейках таблицы устанавливается выравнивание текста по центру по вертикали. По горизонтали текст выравнивается по центру либо по левому краю. Границы ячеек таблицы должны быть только чёрного цвета, толщина линии – 1 пт. На все таблицы должны быть приведены ссылки в тексте статьи, при ссылке следует писать слово «табл.» с указанием её номера, например, « … данные приведены в табл. 5». Таблицы нумеруются в пределах статьи. Таблица располагается сразу после ссылки на неё, если это возможно (например, после окончания абзаца). Если же таблица не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью таблицы, ширина которой превышает ширину колонки. В этом случае таблица и её название размещаются по центру страницы. Таблица не должна выступать за границы полей страницы. Таблица и её название отделяются от основного текста статьи одной пустой строкой до и после.

Рисунки в статье

Ссылки на иллюстрации по тексту статьи обязательны и оформляются в виде « ... на рис. 2» и т. п. Рисунок и его подпись выравниваются по центру колонки (без абзацных отступов), положение рисунка – «в тексте». Размещается рисунок после его первого упоминания в тексте, если это возможно (например, после окончания абзаца). Если же иллюстрация не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью рисунка, ширина которого превышает ширину колонки. В этом случае рисунок и его подпись выравниваются по центру страницы. Иллюстрация не должна выступать за границы полей страницы. Подпись рисунка оформляется в формате: слово «Рисунок», пробел, номер иллюстрации, пробел, тире, пробел,

название рисунка. Название рисунка записывают с прописной буквы без точки в конце строки. Для подписи иллюстрации применяют курсивное начертание. Иллюстрация и её подпись отделяются от основного текста статьи одной пустой строкой до и после. Не допускается выполнять рисунки с помощью встроенного графического редактора Microsoft Office Word. Если на иллюстрации имеется текст, размер шрифта должен быть не менее чем аналогичный текст, набранный шрифтом Times New Roman 10-го размера. Иллюстрация не должна содержать много незаполненного пространства.

Формулы

Формулы и уравнения рекомендуется набирать с использованием MathType (предпочтительно) или MS Equation. Формулы и математические символы не должны существенно отличаться по размеру от основного текста. Обязательной является нумерация формул, на которые имеется ссылка в тексте статьи. Ссылки в тексте на порядковые номера формул дают в скобках, например, « ... согласно формуле (2)». Формулы размещаются по центру колонки, а их номера – по правому краю. Как для строки с формулой, так и для первой строки пояснений (при наличии), абзацный отступ убирается. Первая строка пояснения начинается со слова «где», после которого следует поставить табуляцию на 1 см, затем само пояснение в формате: символ, подлежащий объяснению, пробел, тире, пробел, поясняющий текст, запятая, обозначение единицы измерения физической величины. Пояснения перечисляются через точку с запятой, выравниваются по ширине. Вторая и последующие строки пояснений начинаются с абзацного отступа (1 см). Весь блок текста, связанный с формулой (только формула, несколько формул подряд или формула с пояснениями), отделяется от основного текста одной пустой строкой до и после. Переносить формулы на следующую строку допускается только на знаках выполняемых операций, причем знак в начале следующей строки повторяют. При переносе формулы на знаке умножения применяют знак «×». Формулы и математические уравнений могут быть записаны в тексте документа, если их высота не превышает высоту строки. При этом следует учитывать, что знаки математических операций отделяются от чисел или символов пробелами с обеих сторон. Например, «Если учесть, что $y < 0$ и $2x + y = 1$, то из формулы (3) можно выразить x…». К символам, которые приведены в формуле, при дальнейшем их употреблении (в том числе в пояснениях к формуле) должно применяться курсивное начертание. При этом к любым числам (верхние и нижние индексы, содержащие цифры и т. п.), а также к математическим знакам курсивное начертание не применяется. Не допускается вставлять формулы, выполненные в виде рисунков.

Перечисления: оформление списков

Основной текст статьи может содержать перечисления, оформленные в виде маркированного списка. В качестве маркера элемента списка разрешается использовать только короткое тире «–». Каждый элемент перечисления записывается с новой строки с абзацного отступа, равного 1 см. После символа короткого тире текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов. Нумерованные и многоуровневые списки включать в статью не разрешается.

Литература

В тексте статьи обязательны ссылки на все литературные источники, номер источника указывается в квадратных скобках. Ссылки на неопубликованные работы не допускаются. Рекомендуемое количество источников, на которые ссылается автор, не менее 10. Перечень источников приводится в порядке их упоминания в статье. Библиографическое описание каждого литературного источника оформляется в соответствии с ГОСТ Р 7.0.100–2018. Перечень литературных источников оформляется в виде нумерованного списка. В качестве маркеров элементов списка используют порядковые арабские цифры с точкой. Каждый источник представляет собой отдельный элемент перечисления, записывается с новой строки с абзацного отступа, равного 1 см. После порядкового номера с точкой текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов.

В конце статьи обязательно приводятся аннотации на русском и английском языках, каждая заканчивается перечнем 5-6 ключевых слов.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Слово «Аннотация» опускается. Текст аннотации начинается с ФИО авторов и названия статьи, выделяемых полужирным начертанием. Аннотация на русском языке совпадает с аннотацией, приведенной в начале статьи. В тексте аннотации на английском языке после фамилии автора указывается только первая буква имени с точкой. Абзацный интервал «перед» – 12 пт, «после» – 12 пт. Ключевые слова оформляются с новой строки аналогично тексту аннотации. Заголовок «Ключевые слова:» (англ. «Keywords:») выделяется полужирным начертанием. Ключевые слова перечисляются через запятую.

Порядок представления статьи и сопроводительные документы

В редакцию необходимо представить:

– файл с текстом статьи;

– файл, содержащий фамилию, имя и отчество авторов полностью; ученую степень, ученое звание; место работы с полным указанием должности, подразделения и наименования организации, города (страны); номера телефонов и e-mail для связи;

– экспертное заключение о возможности публикации статьи, подписанное руководителем и заверенное печатью организации, в которой работает автор статьи;

– выписка из заседания кафедры или письмо организации с просьбой об опубликовании и указанием, что изложенные в статье результаты ранее не публиковались.

Статьи и сопроводительные документы следует высылать на электронный адрес infcyb.donntu@yandex.ru.

К сведению авторов

Если статья оформлена с нарушением указанных выше требований и правил, редакция после предварительного рассмотрения может отклонить статью.

На рецензирование статьи направляются членам редакционной коллегии журнала. Все статьи публикуются при наличии положительной рецензии.

В статью могут быть внесены изменения редакционного характера без согласования с автором. Ответственность за содержание статьи и качество перевода аннотаций несут авторы.

Публикация статей в научном журнале «Информатика и кибернетика» осуществляется на некоммерческой основе.

Все номера Научного журнала размещаются в электронной библиотечной системе ФГБОУ ВО «ДонНТУ» и на сайте http://infcyb.donntu.ru/.

CONTENT

Informatics and computer engineering

Электронное периодическое издание

Научный журнал

ИНФОРМАТИКА И КИБЕРНЕТИКА

(на русском, английском языках)

№ 1 (35) - 2024

Ответственный за выпуск Р. В. Мальчева

Технический редактор Р. В. Мальчева

Компьютерная верстка Р. В. Мальчева

Подписано к выпуску 18.04.2024. Усл. печ. лист. 7,7. Уч.-изд. лист. 4,2.
Адрес редакции: ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36, ул. Кобозева, 17.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
E-mail: infcyb.donntu@yandex.ru, URL: http://infcyb.donntu.ru