DOI: 10.18137/RNU.V9187.23.03.P.231 УДК 004.42:004.9

С.А. Чернышев

ПРОБЛЕМЫ МУЛЬТИАГЕНТНЫХ СИСТЕМ И ВОЗМОЖНЫЕ ПУТИ ИХ РЕШЕНИЯ

Аннотация. Рассматривается история развития мультиагентных систем, освещаются их основные преимущества. Обсуждаются причины, из-за которых данная технология не достигла распространения: сложность BDI-модели агентов, стандарты FIPA, отсутствие промышленных методологий разработки мультиагентных систем и др. Предлагаются различные варианты решений выхода из сложившейся ситуации: формирование библиотеки общих шаблонов проектирования мультиагентных систем, адаптация существующих подходов и технологий из области программной инженерии для разработки архитектур агента и мультиагентной системы в целом, использование больших языковых моделей (LLaMa, Alpaca и др.) для реализации интеллектуальных программных агентов и др.

Ключевые слова: мультиагентные системы, парадигма мультиагентных систем, проблемы, перспективы.

S.A. Chernyshev

PROBLEMS OF MULTI-AGENT SYSTEMS AND POSSIBLE WAYS TO SOLVE THEM

Abstract. This article discusses the history of the development of multi-agent systems, highlights their main advantages, and addresses the reasons why this technology has not become widespread: the complexity of the BDI agent model, FIPA standards, the lack of industrial of multi-agent systems development methodologies, etc. Various solutions are proposed to overcome the current situation, such as: creating a library of common of multi-agent systems design patterns, adapting existing approaches and technologies from the field of software engineering for the development of agent architectures and the overall multi-agent system, and using large language models (LLaMa, Alpaca, etc.) for the implementation of intelligent software agents, etc.

Keywords: multi-agent systems, multi-agent systems paradigm, problems, perspectives.

Введение

В середине 1980-х годов научному и индустриальному сообществу была предложена концепция мультиагентных систем (далее – МАС). Она была высоко оценена, что положило начало множеству научных исследований, в которые было вовлечено достаточно большое количество людей. Это позволило за несколько десятилетий сформировать базовые теоретические основы МАС, различные методологии их проектирования, такие как Gaia [1], MaSE [2], ADELFE [3] и др., а также инструментальные средства их поддержки: agentTool [4], MASDK [5], agentBuilder [6] и ряд других. Активно велись работы по различным моделям агентов: BDI (Belief – Desire – Intention, убеждение – желание – намерение), логические, реактивные, проактивные, самоорганизующиеся, поведенческие и др.

К началу 21-го века была создана общественная организация FIPA (Foundation for Intelligent Physical Agents), на которую возлагались большие надежды в области разработки стандартов для интеллектуальных агентов и МАС. С ее преобразованием в один из комитетов по стандартизации IEEE в 2005 году казалось, что парадигма проектирования сложных адаптивных систем индустриального уровня с использованием мультиагентных

Чернышев Станислав Андреевич
кандидат технических наук, доцент кафедры информатики, Санкт-Петербургский государственный экономический университет; доцент кафедры прикладной информатики, Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург. Сфера научных интересов: имитационное моделирование, мультиагентные системы, интеллектуальные системы поддержки принятия решений, разработка распределенных систем. Автор более 40 опубликованных научных работ.
Электронный адрес: chernyshev.s.a@bk.ru

технологий займет лидирующие позиции и задаст совершенно новый вектор развития программной инженерии и искусственного интеллекта.

Отчасти это было связано с тем, что МАС как технология показывала зачатки зрелости; особенно привлекательной выглядела сама концепция мультиагентных систем и возможности, которые она открывала. С самых первых дней парадигма МАС была ориентирована на создание сложных адаптивных систем с использованием биологических принципов, где задачи предлагалось решать путем взаимодействия агентов (общение друг с другом), способных к самоорганизации. Другими словами, сложная задача разбивается на ряд простых подзадач, решение которых берут на себя программные автономные агенты, способные работать в различных режимах (параллельно или асинхронно) и взаимодействовать между собой с использованием протокола обмена сообщениями, который напоминает естественный язык. Агенты также могут генерировать некоторые события, оценивать свое состояние, передавать данные в виртуальную среду, синхронизироваться с другими агентами в рамках коллективных действий и др. Такие свойства агентов позволяют при проектировании МАС довольно эффективно решать проблему по взаимодействию множества компонентов программы между собой. Именно поэтому большое количество научных работ посвящены различным способам взаимодействия между программными агентами на основе протоколов и диалогов (вычисление на основе взаимодействий), а их наработки начали использоваться в различных программных архитектурах и технологиях разработки сложных интеллектуальных систем.

В 2005 году были подведены итоги за 20 лет существования и развития парадигмы проектирования МАС и дана оценка ее перспективности [7]. Авторы прогнозировали, что количество индустриальных внедрений МАС до 2015 года увеличится до 250. Но эти прогнозы были сильно преувеличены, так как по итогам анализа, приведенного в работах [8; 9], за этот период было лишь 46 индустриальных внедрений. Несмотря на те преимущества, которыми обладала парадигма МАС, интерес инвесторов и промышленности, ИТ-корпораций к ней охладел, и ее место заняли такие технологии, как сервис и событийно-ориентированная архитектура, ГРИД-вычисления, облачные и туманные вычисления, микросервисы и др.

Некоторые ученые связывают такое положение дел с тем, что технология МАС недостаточно зрелая, приводя в пример то, сколько времени потребовалось объектно-ориентированному программированию до того момента, пока оно не приобрело практическую значимость (чуть более 30 лет). Другие сетуют на слабость существующих методологий проектирования МАС и поддерживающих их инструментов. Третьи заявляют, что пора

отходить от концепции, что МАС можно использовать в каждой области для сужения предметных областей, где у технологии не будет значимых конкурентов. К таким направлениям относятся сложные, многокритериальные, вычислительно затратные в реальном режиме времени и плохо формализуемые задачи, в решении которых МАС уже показали свои неоспоримые преимущества: извлечение знаний, распознавание образов, управление ресурсами, планирование расписаний и др. [8].

Существующие проблемы

BDI-модель агентов и МАС. Одной из самых первых моделей программного автономного агента была BDI-модель, где поведение агента, его ментальные состояния и распределенное взаимодействие агентов предлагалось описывать в терминах предикатов первого порядка [9]. У каждого агента должны быть убеждения (знания о внешнем мире), цели (состояния, к которым агент должен стремиться), и он должен тем или иным образом реагировать на события, происходящие в окружающем его мире, основываясь на своих убеждениях и целях. Намерения агента могут быть как индивидуальными, так и коллективными (обязательства) и должны быть направлены на достижение целей. Проблема заключается в том, что использование предикатов первого порядка не позволяет описать все возможные случаи языком классической логики, а сама программная реализация агентов на основе BDI-модели настолько сложна и противоречит парадигме МАС (вычисление на основе взаимодействий), что обычные программисты обходили и обходят ее стороной. Так как это была одна из первых работ, то большое количество ученых брали ее за основу при проведении собственных исследований и разработки новых методологий проектирования МАС, которые оказывались еще сложнее в понимании и реализации, чем BDI-модель. Такое положение дел отрицательно сказалось на мультиагентных системах и, как заявлено в работе [8], «затормозило практическое использование МАС технологий как минимум на десятилетие».

Стандарты и деятельность FIPA. С самого первого для существования FIPA в него входили ученные продвигающие BDI-модель, что в целом сказалось на предлагаемых в этих стандартах подходах. Особенно это касается предложенного языка коммуникации между агентами ACL (Agent Communication Language) [11]. Он очень сложен для понимания рядовым программистом и описания сообщений, которые будут использоваться агентами в процессе коммуникации друг с другом. Дополнительно к этому спецификация стандартов FIPA не ориентирована на параллельные вычисления, а это противоречит самой парадигме МАС. Отчасти это связано с тем, что не было более плотного сотрудничества FIPA с индустриальными стейхолдерами.

Как показывает анализ [8], индустриального внедрения удалось добиться МАС, разработчики которых отошли от той модели взаимодействия между агентами, которую предлагает FIPA, и реализовали собственные решения протоколов взаимодействия. С такой же проблемой столкнулся и коллектив под руководством автора при разработке некоторых МАС [12–14], и только отойдя от стандартов FIPA, реализовав собственный протокол взаимодействия между агентами на основе формата JSON и среду выполнения агентов, две из трех работ удалось довести до индустриального внедрения.

Отсутствие промышленных методологий разработки МАС. Существует огромное количество методологий МАС, но практически все они, как и поддерживающие их инструменты, разработаны учеными. Это сказалось на том, что бо́льшая часть разработанных за 30 лет методологий не ушла дальше статей, некоторые из них были поддержа-

ны инструментарием, но так и не добрались до полноценного применения. Наибольших успехов удалось достичь таким методология, как Gaia, MaSE, ADELFE и др. Они основаны на базе методологии Model-Driven Development (MDD, разработка, управляемая моделями), которая используется при разработке программных продуктов и призвана максимально сократить время, выделяемое разработчиком на кодирование. В основе MDD лежит разработка модели, изначально не привязанной к конкретной платформе или программным инфраструктурам. Модель описывается посредством программного обеспечения, позволяющего абстрагироваться от кодового представления предметной области (например, UML), после чего проходит через этап кодогенерации, и разработчики добавляют код там, где это необходимо, вручную.

Несмотря на наличие многих статей в период с 2005 по 2015 год [8], где в основе предлагаемой методологии проектирования МАС использовалось MDD, они не получили своего последующего развития и применения на практике.

Это связано с недостатками MDD, к которым относятся:

• увеличение финансовых затрат из-за необходимости поддерживать и использовать специализированное программное обеспечение;

• не все программисты способны вести разработку или имеют необходимый набор знаний для проектирования на основе диаграмм;

• в сгенерированном коде может быть огромное количество зависимостей, которые придется удалять вручную или принять решение об их поддержке на всем этапе жизненного цикла ПО.

Дополнительно к недостатку практически всех методологий проектирования МАС можно отнести то, что они были направлены на научное сообщество, а не на разработчиков программного продукта. И так как в основе большинства методологий лежала BDI-модель агента, это в разы усложняло разработку и сопровождение программных инструментов поддержки методологий. Отсюда – сложность их продвижения, отказ сообщества поддерживать инструментарий данных методологий с открытым исходным кодом, вследствие чего наблюдается угасание интереса к самой парадигме МАС. Возможно, это связано также с тем, что за всё время существования методологий МАС не удалось добиться значимых успехов (прорывов) в области индустриального внедрения.

Различное толкование ключевых понятий МАС. В работе [15] обозначен вопрос различного понимания разработчиками и учеными существующих проблем в парадигме МАС, вследствие чего усложняется взаимопонимание между ними, или оно становится полностью невозможным, так как отсутствуют соглашения по основным понятиям в области МАС.

Позиционирование МАС научным сообществом. С самых первых дней появления парадигмы МАС она позиционировалась как некая «серебряная пуля», способная решить все существующие проблемы, которую стоит использовать в любом новом проекте. К сожалению, история IT-индустрии помнит много таких случаев, когда появлялся новый язык программирования или технология, которым все пророчили успех и универсальность применения, но в итоге про них уже давно никто не вспоминает. Обычно такое позиционирование отталкивает разработчиков и заставляет с большей осторожностью и скепсисом смотреть на возможности технологии. Так получилось и с мультиагентными системами. Очень много ниш, которые рассматривались как потенциальные для их применения, были заняты более элегантными, простыми в разработке и сопровождении ин-

дустриальными решениями (сервис и событийно-ориентированная архитектура, ГРИД-вычисления и др.).

Сложность проектирования и внедрения. Ни одна из существующих методологий проектирования программного обеспечения не сравнится по сложности с проектированием мультиагентных систем. Так, например, при планировании сил и времени на разработку необходимо было в 3–5 раз больше, чем изначально закладывалось [16; 17]. Минимальный срок первого прототипа системы в лучшем случае составляет от 3 до 6 месяцев, а внедрение занимает в несколько раз больше времени, чем было потрачено на разработку. Это связано с тем, что необходимо дополнять онтологию предметной области, отрабатывать правила принятия и согласования решений агентами и, конечно, интегрировать разработанную МАС с информационными системами стейкхолдеров.

Даже если ранее была уже разработана аналогичная мультиагентная система, то из-за специфики бизнес-процессов нового предприятия процесс внедрения занимает обычно значительно больше времени.

К тому же повторный запуск МАС с теми же параметрами и входными данными может дать совсем другой результат. Это связано как с недетерминизмом параллельных асинхронных процессов [8], так и с наличием малых флуктуаций решений или вероятностных механизмов в алгоритме выбора эвристик, которые используются для поиска решения, приближенного к наилучшему из возможных вариантов. Данную проблему можно обойти путем запуска сразу нескольких (10–20) копий МАС с последующим выбором наилучшего решения, которое чаще всего будет постоянным. Но это потребует больше вычислительных ресурсов [18].

Отсутствие базы данных общих программных шаблонов проектирования МАС и их классификации. В том случае, когда стейкхолдер не предъявляет требования к отсутствию зависимостей у разрабатываемого программного продукта от мультиагентных фреймворков или сторонних библиотек, процесс разработки протекает значительно легче и быстрее. Это связано с тем, что фреймворки или среды разработки имитационных моделей уже реализуют необходимый функционал среды запуска агентов, коммуникацию между ними, организацию доступа к ресурсам и др. Среди таких инструментов наиболее выделяются следующие: Jade, Anylogic, Gamma, NetLogo, Repast Symfony. Они же используются и при обучении проектированию мультиагентных систем в большинстве учебных заведениях. К сожалению, эти инструменты «навязывают» свои правила игры разработчику: архитектурный подход, часть готовых решений, сервисов и др. В среднесрочной перспективе в этом нет ничего плохого, но когда стоит задача разработать полноценную мультиагентную систему с нуля без наличия зависимостей от существующих мультиагентных фреймворков или инструментарий, а также внести фундаментальные (или не очень) изменения в архитектуру существующей системы, у студента, инженера или исследователя наступает ступор, связанный с незнанием подходов и архитектурных паттернов (шаблонов), лежащих в основе мультиагентных фреймворков.

У разработчиков программного обеспечения имеется масса различных шаблонов проектирования: GoF, GRASP и др., что позволяет им не искать каждый раз решение той или иной архитектурной проблемы в процессе проектирования, а адаптировать уже имеющиеся под кодовую базу проекта.

Наличие аналогичной базы по шаблонам проектирования, не привязанных к конкретным предметным областям, которые можно использовать при проектировании МАС лю-

бой сложности, и классификация представленных в ней шаблонов позволят разработчикам решать типовые проблемы, возникающие в процессе проектирования и кодирования МАС, а также общаться ученым и инженерам на одном зыке.

Возможные пути решения

Первым делом необходимо избавить парадигму МАС от того, что ее длительное время позиционировали как универсальную технологию, способную решать любые индустриальные задачи, четко обозначив спектр применения технологии. Именно из-за такого изначального отношения ученных только спустя более 30 лет с появления парадигмы МАС начинает формироваться понимание, в каких предметных областях имеются неоспоримые преимущества: интернет вещей, умный дом, извлечение знаний, распознавание образов, управление ресурсами, планирование расписаний и др. Но и здесь не все так хорошо, как хотелось бы, так как многие из этих направлений уже заняты существующими на рынке технологиями, с которыми достаточно сложно конкурировать ввиду их более низкой стоимости и сложности внедрения. Так, например, интересными представляются работы по использованию МАС в аэрокосмической отрасли [19; 20], разрешении сложных конфликтных ситуаций на предприятии [21], адаптивном управлении ресурсами [22], системе «Умный город» [23] и др.

Также необходимо пересмотреть подход к проектированию программной архитектуры агента и мультиагентной системы в целом, не придумывая что-то совершенно новое, а адаптировав актуальные виды архитектур из области программной инженерии, как например: гексагональная, чистая и др.; использовать уже готовые решения: брокеры сообщений (например, Kafka), системы виртуализации и кластеризации, способные ускорить стадию разработки МАС, повысив удобство последующего сопровождения и добавления нового функционала в агенты или саму МАС.

Методология проектирования МАС должна носить гибкий характер, оставляя последнее слово за программным архитектором, задавать общие, абстрактные контуры и направление проектирования МАС, а не конкретные архитектурные решения, которые насколько хорошими бы ни были, всё равно не подойдут для всех предметных областей. Такая методология может быть основана на уже существующих и зарекомендовавших себя методологиях проектирования программных продуктов: Feature-Driven Development (разработка, управляемая функциональностью), Domain-Driven Development (предметно-ориентированное проектирование) или Behavior-Driven Development (разработка через реализацию поведения). Адаптация актуальной в индустрии методологии под задачи проектирования МАС позволит задействовать уже существующий инструментарий поддержки этой методологии, а главное – использовать уже сформированные лучшие практики. Как из мультиагентных систем в свое время было заимствовано много концептуальных идей, которые используются в таких современных технологиях, как сервис и событийно-ориентированная архитектура, ГРИД-вычисления, облачные и туманные вычисления, микросервисы, так ничто не мешает позаимствовать их лучшие практики при проектировании нового поколения МАС. К тому же текущие наработки в генеративных нейронных сетях (GPT-4, LLaMa, Alpaca и др.) позволяют задавать сценарии поведения и реакции на те или иные входные данные при реализации интеллектуальных программных агентов.

Что касается формирования базы шаблонов проектирования и их классификации, то эта работа уже проделана автором [24; 25]. Из более чем 200 шаблонов проектирования,

представленных в различных источниках, удалось отобрать 60, которые не привязаны ни к одной предметной области и могут быть использованы при проектировании программных автономных агентов и МАС различной сложности, а также предложена их следующая классификация, упрощающая поиск шаблона для решения возникающих проблем в процессе проектирования:

- структурные;
- поведенческие;
- миграционные;
- коммуникационные;
- архитектурные (системные);
- защитные;
- когнитивные.

Некоторые из классов шаблонов позволяют ввести в систему дополнительные элементы, расширяющие ее функциональные возможности, в то время как другие направлены на реализацию различных аспектов работы как агента, так и МАС. Из всех предложенных классов шаблонов наиболее выделяются архитектурные (системные), так как они задают различные виды архитектур агентов, мультиагентных систем или элементов, которые закладывают жесткие программные ограничения функционирования разрабатываемой системы или ее частей.

Заключение

Несмотря на то, что за прошедшие 10 лет мультиагентные системы потеряли свои позиции и не упоминались на различных презентациях из-за сложившегося в индустриальной среде отрицательного имиджа, даже если и использовались такими гигантами, как Google, Yandex, IBM, постоянно возрастающая сложность проектируемых систем и тренд на их адаптивность, самоорганизацию и интеллектуализацию заставляет разработчиков программного обеспечения, ученых и инженеров всё чаще обращаться к наработкам в области МАС. Это требует от них определенного набора компетенций в области объектно-ориентированного программирования, алгоритмов и структур данных, параллельного и асинхронного программирования, шаблонов и принципов проектирования программных продуктов, искусственного интеллекта и др.

Количество индустриальных внедрений МАС постепенно увеличивается, но, конечно, не теми темпами, которые прогнозировались в начале XXI века. В связи с этим перед научным и инженерным сообществом стоят весьма непростые задачи: переосмысление парадигмы мультиагентных систем и ее адаптация к современным реалиям и потребностям бизнеса. Необходимо формализовать описание моделей агентов и МАС, пересмотреть программную архитектуру и подходы к реализации автономных интеллектуальных агентов и мультиагентных систем, по-новому взглянуть на процессы взаимодействия между агентами и предложить более современные способы общения с учетом их распределенной работы, адаптировать современные методологии разработки программных продуктов по потребности МАС, что положительно скажется на привлечении в эту область рядовых разработчиков, так как работать придется с известным инструментарием и стеком технологий, предложить и разработать более современный инструментарий для создания МАС (с использованием больших языковых моделей).

Другими словами, мультиагентные системы как технология прошли период завышенных ожиданий и находятся на ранней стадии преодоления недостатков, когда научное и

инженерное сообщество осознает, какие шаги при становлении парадигмы МАС были ошибочны, к каким последствиям они привели и в каком направлении необходимо двигаться для исправления данной ситуации.

Литература

1. *Zambonelli F., Jennings N., Wooldridge* M. (2003) Developing Multiagent systems: The GAIA methodology. ACM Transactions on Software Engineering and Methodology, 2003, No. 12, Pp. 417–470.
2. *Deloach S.* (2001) Analysis and Design using MaSE and agentTool. Proc. of the 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS), Miami University Press, 2001.
3. *Bernon C.* (2002) ADELFE: A Methodology for Adaptive Multi-Agent Systems Engineering. Proc. of the 3th International Workshop on Engineering Societies in the Agents World, 2002, Pp. 156–169.
4. *Garcia-Ojeda J.* (2009) AgentTool Process Editor: Supporting the Design of Tailored Agent-based Processes. Proc. of the 24th Annual ACM Symposium on Applied Computing, March 8–12, 2009, Honolulu, Hawaii, USA.
5. *Gorodetsky V., Karsaev O., Samoylov V.* (2009) Support for Analysis, Design and Implementation Stages with MASDK. Eds.: Luck M., Gomez-Sanz J.J. Springer, Heidelberg, LCNS, 2009, Vol. 5386, Pp. 272–288.
6. Инструментальные средства для открытых сетей агентов / В.И. Городецкий, О.В. Карсаев, В.В. Самойлов [и др.] // Известия РАН. Теория и системы управления. 2008. № 3. С. 106–124.
7. *Luck M.* (2005) Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing). AgentLink. URL: http://www.agentlink.org/roadmap/ (дата обращения: 20.10.2022).
8. *Городецкий В.И., Бухвалов О.Л., Скобелев П.О., Майоров И.В.* Современное состояние и перспективы индустриальных применений многоагентных систем // Управление большими системами. 2017. Выпуск 66. С. 94–157. DOI: https://doi.org/10.25728/ubs.2017.66.4
9. *Műller J., Fisher K.* (2013) Application Impact of Multiagent Systems and Technologies: A Survey. Agent-Oriented Software Engineering» book series. Springer, 2013, Pp. 1–2.
10. *Wooldridge M.* (2009) An Introduction to MultiAgent Systems. John Wiley & Sons, 2009, 368 p.
11. ACL Agent Communication Language.
12. URL: http://fipa.org/specs/fipa00061/SC00061G.pdf (дата обращения: 21.10.2022).
13. Свидетельство о государственной регистрации программы для ЭВМ № 2022662419 от 04.07.2022. г. Мультиагентная система поддержки принятия решений организации логистических цепочек / Чернышев С.А., Антонов А.А., Дук Г.В.
14. Свидетельство о государственной регистрации программы для ЭВМ № 2022617716 от 25 апреля 2022 г. Мультиагентная система поддержки принятия решений для формирования расписания обучения в вузе / Чернышев С.А., Нагорных М.Э., Быков А.Н.
15. Свидетельство о государственной регистрации программы для ЭВМ № 2022619068 от 19 мая 2022 г. Мультиагентная система поддержки принятия решений для минимизации стоимости группируемых товаров на основе реактивных агентов / Чернышев С.А., Магомедов О.Р.
16. *Deloach S.A* (2009) Moving multi-agent systems from research to practice. Int. J. Agent-Oriented Software Engineering, 2009, Vol. 3, No. 4, Pp. 378–382.
17. *Виттих В.А., Скобелев П.О.* Мультиагентные модели взаимодействия для построения сетей потребностей и возможностей в открытых системах // Автоматика и телемеханика. 2003. № 1. С. 177–185.

18. *Скобелев П.О.* Интеллектуальные системы управления ресурсами в реальном времени: принципы разработки, опыт промышленных внедрений и перспективы развития // Приложение к теоретическому и прикладному научно-техническому журналу «Информационные технологии». 2013. № 1. С. 1–32.
19. *Магомедов О.Р., Чернышев С.А.* Мультиагентная система поддержки принятия решений для минимизации стоимости группируемых товаров // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ и управление. 2022. Вып. 4. С. 97–107. DOI: 10.18137/RNU.V9187.22.04.P.97
20. *Галузин В.А.* Разработка моделей, методов и средств создания цифровой платформы согласованного планирования целевого применения гетерогенных группировок малых космических аппаратов дистанционного зондирования земли // Вестник Самарского государственного технического университета. Серия: Техническиенауки. 2022. № 1 (73). С. 20–45. DOI: 10.14498/tech.2022.1.2
21. *Skobelev P., Galuzin V., Galitskaya A., Travin V.* (2021) Intelligent system for real-time adaptive management of groups of small satellites: design and experimenting. Journal of Physics: Conference Series 13: Multiconference on Control Problems, MCCP 2020. P. 012085. DOI: 10.1088/1742-6596/1864/1/012085
22. *Швецов А.Н., Дианов С.В., Дианов Д.С.* Проектирование мультиагентной системы разрешения межфункциональных конфликтов на предприятии // Вестник Череповецкого государственного университета. 2022. № 1 (106). С. 74–89. https://doi.org/10.23859/1994-0637-2022-1-106-6
23. *Grachev S.P., Zhilyaev A.A., Laryukhin V.B.* (2021) Methods and Tools for Developing Intelligent Systems for Solving Complex Real-Time Adaptive Resource Management Problems. Autom Remote Control 82, 1857–1885 (2021). https://doi.org/10.1134/S0005117921110035
24. *Kozhevnikov S., Svitek M., Skobelev P.* (2021) MULTI-AGENT APPROACH FOR SMART RESILIENT CITY. Studies in Computational Intelligence, 2021, Vol. 952, Pp. 215–227. DOI: 10.1007/978-3-030-69373-2_15
25. *Чернышев С.А.* Классификация общих шаблонов проектирования мультиагентных систем // Программные продукты и системы. 2022. Т. 35, № 4. С. 670–679. DOI: 10.15827/0236-235X.140.670-679
26. *Чернышев С.А.* Общие программные шаблоны проектирования мультиагентных систем: монография. Москва: РУСАЙНС, 2022. 134 с.

Literature

1. Zambonelli F., Jennings N., Wooldridge M. (2003) Developing Multiagent systems: The GAIA methodology. ACM Transactions on Software Engineering and Methodology, 2003, No. 12, Pp. 417–470.
2. Deloach S. (2001) Analysis and Design using MaSE and agentTool. Proc. of the 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS), Miami University Press, 2001.
3. Bernon C. (2002) ADELFE: A Methodology for Adaptive Multi-Agent Systems Engineering. Proc. of the 3th International Workshop on Engineering Societies in the Agents World, 2002, Pp. 156–169.
4. Garcia-Ojeda J. (2009) AgentTool Process Editor: Supporting the Design of Tailored Agent-based Processes. Proc. of the 24th Annual ACM Symposium on Applied Computing, March 8–12, 2009, Honolulu, Hawaii, USA.
5. Gorodetsky V., Karsaev O., Samoylov V. (2009) Support for Analysis, Design and Implementation Stages with MASDK. Eds.: Luck M., Gomez-Sanz J.J. Springer, Heidelberg, LCNS, 2009, Vol. 5386, Pp. 272–288.

6. Gorodetsky V.I., Karsaev O.V., Samoilov V.V. (2008) [Tools for Open Agent Networks]. *Izvestija RAN. Teorijaisistemyupravlenija*, 2008, No. 3, Pp. 106–124 (in Russian).
7. Luck M. (2005) Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing). AgentLink. URL: http://www.agentlink.org/roadmap/ (accessed: 20.10.2022).
8. Gorodetsky V., Bukhvalov O., Skobelev P., Mayorov I. (2008) [Industrial applications of multi-agent systems: current state and prospects]. Upravlenie bol'shimi sistemami, 2008, Iss. 66, No. 3, Pp. 106–124. DOI: https://doi.org/10.25728/ubs.2017.66.4 (in Russian).
9. Műller J., Fisher K. (2013) Application Impact of Multiagent Systems and Technologies: A Survey. Agent-Oriented Software Engineering book series. Springer, 2013, Pp. 1–2.
10. Wooldridge M. (2009) An Introduction to MultiAgent Systems. John Wiley & Sons. 368 p.
11. ACL Agent Communication Language.
12. URL: http://fipa.org/specs/fipa00061/SC00061G.pdf (accessed: 21.10.2022).
13. Chernyshev S. A., Antonov A. A., Duk G. V. Multi-agent decision support system for the organization of supply chains: Certificate of state registration of the computer program No. 2022662419 dated 04.07.2022 (in Russian).
14. Chernyshev S.A., Nagornykh M.E., Bykov A.N. Multi-agent decision support system for the formation of the training schedule at the university: Certificate of state registration of the computer program No. 2022617716 dated April 25, 2022 (in Russian).
15. Chernyshev S.A., Magomedov O.R. Multi-agent decision support system for minimizing the cost of grouped goods based on reactive agents: Certificate of state registration of a computer program No. 2022619068 dated May 19, 2022 (in Russian).
16. Deloach S.A (2009) Moving multi-agent systems from research to practice. Int. J. Agent-Oriented Software Engineering, 2009, Vol. 3, No. 4, Pp. 378–382.
17. Vittikh V.A., Skobelev P.O. (2003) [Multi-agent models of interaction for building networks of needs and opportunities in open systems]. *Avtomatika i telemehanika*, 2003, No. 1. Pp. 177–185 (in Russian).
18. Skobelev P.O. (2013) [Intelligent resource management systems in real time: development principles, industrial implementation experience and development prospects]. *Prilozhenie k teoreticheskomu i prikladnomu nauchno-tehnicheskomu zhurnalu «Informacionnye tehnologii»*, 2013, No. 1, Pp. 1–32 (in Russian).
19. Magomedov O.R., Chernyshev S.A. (2022) [Multi-agent decision support system for minimizing the cost of grouped goods]. *Vestnik Rossijskogo novogo universiteta. Serija: Slozhnye sistemy: modeli, analiz i upravlenie*, 2022, No. 4, Pp. 97–107. DOI: 10.18137/RNU.V9187.22.04.P.97 (in Russian).
20. Galuzin V.A. (2022) [Development of models, methods and tools for creating a digital platform for agreed planning of heterogeneous group of small space satellites for remote earth sensing]. *Vestnik Samarskogo gosudarstvennogo tehnicheskogo universiteta. Serija: Tehnicheskie nauki*, 2022, No. 1, Pp. 20–45. DOI: 10.14498/tech.2022.1.2 (in Russian).
21. Skobelev P., Galuzin V., Galitskaya A., Travin V. (2021) Intelligent system for real-time adaptive management of groups of small satellites: design and experimenting. Journal of Physics: Conference Series 13: Multiconference on Control Problems, MCCP 2020. P. 012085. DOI: 10.1088/1742-6596/1864/1/012085
22. Anatoly N. Shvetcov, Sergey V. Dianov, Daniil S. Dianov (2022) [Designing a multi-agent system for resolving interfunctional conflicts in an enterprise]. *Vestnik Cherepoveckogo gosudarstvennogo universiteta*, 2022, No. 1, Pp. 74–89. https://doi.org/10.23859/1994-0637-2022-1-106-6 (in Russian).
23. Grachev S.P., Zhilyaev A.A., Laryukhin V.B. (2021) Methods and Tools for Developing Intelligent Systems for Solving Complex Real-Time Adaptive Resource Management Problems. Autom Remote Control 82, 1857–1885 (2021). https://doi.org/10.1134/S0005117921110035

24. Kozhevnikov S., Svitek M., Skobelev P. (2021) MULTI-AGENT APPROACH FOR SMART RESILIENT CITY. Studies in Computational Intelligence, 2021, Vol. 952, Pp. 215–227. DOI: 10.1007/978-3-030-69373-2_15
25. Chernyshev S.A. (2022) [Classification of common design patterns for multi-agent systems]. *Programmnye produkty i sistemy*, 2022, Vol. 35, No. 4, Pp. 670–679. DOI: 10.15827/0236-235X.140.670-679 (in Russian).
26. Chernyshev S.A. (2022) *Obshchiye programmnyye shablony proyektirovaniya mul'tiagentnykh sistem: monografiya* [General software design patterns for multi-agent systems: monograph]. Moscow, RUSSIGNS Publishing, 2022, 134 p. (in Russian).