На правах рукописи

КОВАЛЁВ Сергей Протасович

ТЕОРЕТИКО-КАТЕГОРНЫЕ МОДЕЛИ И МЕТОДЫ ПРОЕКТИРОВАНИЯ БОЛЬШИХ ИНФОРМАЦИОННО-УПРАВЛЯЮЩИХ СИСТЕМ

Специальность: 05.13.17 – Теоретические основы информатики

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
доктора физико-математических наук

Москва – 2013

Работа выполнена в Федеральном государственном бюджетном учреждении науки Институт проблем управления им. В.А. Трапезникова Российской академии наук

Научный консультант:	академик РАН, доктор физико-математических наук Васильев Станислав Николаевич
Официальные оппоненты:	Артамонов Вячеслав Александрович, доктор физико-математических наук, профессор, ФГБОУ ВПО Московский государственный университет им. М.В. Ломоносова, профессор кафедры высшей алгебры механико-математического факультета
	Жданов Александр Аркадьевич, доктор физико-математических наук, профессор, ОАО Институт точной механики и вычислительной техники им. С.А. Лебедева Российской академии наук, главный научный сотрудник
	Кольцов Петр Петрович, доктор технических наук, ФГБУН Научно-исследовательский институт системных исследований Российской академии наук, заместитель директора
Ведущая организация:	ФГБУН Институт динамики систем и теории управления Сибирского отделения Российской академии наук, г.Иркутск

Защита диссертации состоится «___» _____ 2014 г. в __ часов на заседании диссертационного совета Д 002.017.02 при Федеральном государственном бюджетном учреждении науки Вычислительный центр им. А.А. Дородницына Российской академии наук по адресу: 119333, г.Москва, ул.Вавилова, д.40, конференц-зал.

С диссертацией можно ознакомиться в библиотеке Федерального государственного бюджетного учреждения науки Вычислительный центр им. А.А. Дородницына Российской академии наук.

Автореферат разослан «___» _____ 20___ г.

Ученый секретарь
Диссертационного совета Д 002.017.02
д.ф.-м.н., профессор В.В. Рязанов

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность работы. Глобализация экономики и коммуникаций приводит к росту масштаба объектов, требующих охвата единым процессом управления или тесно согласованным комплексом процессов. Например, укрупняются до транснационального и межгосударственного уровня цепи поставки продукции, транспортные инфраструктуры высокой доступности, программы развития отраслей. Давно известно, что к числу основных способов повышения эффективности управления большими объектами относится комплексная автоматизация. Поэтому объекты оснащаются цифровыми измерительными приборами и исполнительными механизмами, которые подключаются к программным средствам управления – информационно-управляющим системам, достигающим не только большого масштаба (large scale), но и сверхбольшого (ultra-large scale systems, ULS)[1]. По сравнению с традиционными системами, они обладают очень большими значениями показателей размера – количества данных, элементов, взаимосвязей, процессов, нормативов, пользователей и др. Примером служит Smart Grid («умная» сеть) – система сквозной автоматизации крупной электроэнергетической сети[2].

Традиционные подходы к созданию информационных систем не были рассчитаны на поддержку больших значений размера. Кроме того, рост масштаба приводит к проявлению принципиально новых проблем, незаметных при малых размерах и затрудняющих соблюдение классических принципов разработки АСУ, сформулированных полвека назад[3]. Например, принцип первого руководителя требует создавать систему под непосредственным руководством лица, способного выступить главным заказчиком (приобретателем) системы и непререкаемым арбитром в разрешении конфликтов между ожиданиями групп пользователей. Однако на большом объекте такое лицо может отсутствовать, если каждый руководитель имеет недостаточный уровень полномочий и/или управляет только частью объекта. К тому же, большой объект нестабилен: почти все время в нем присутствуют участки, находящиеся в процессе существенного изменения, способного повлиять на потребности пользователей. В результате не удается выдать разработчикам полный непротиворечивый набор требований к информационно-управляющей системе, и она входит в непрерывный процесс развития и адаптации. Возникают трудности в применении принципа типовости: типовые решения, предназначенные для автоматизации заранее заданных задач, требуют огромных затрат на адаптацию к априори неизвестным и постоянно меняю-

[1] Ultra-Large-Scale Systems: The Software Challenge of the Future. Pittsburgh: Carnegie Mellon Software Engineering Institute, 2006.

[2] *Anvaari M. et al.* Smart Grid software applications as an ultra-large-scale system: challenges for evolution // Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES. P. 1–6.

[3] *Глушков В.М.* Введение в АСУ. Киев: Техника, 1972.

щимся условиям их использования в системе. Ключевую роль приобретает трассирование компонентов к задачам – одна из самых трудоемких операций в инженерии информационных систем[4].

В связи с этим большой интерес вызывают новые технологии разработки систем, направленные на уменьшение затрат труда путем построения широкого набора моделей, заполняющих «когнитивную дистанцию» между автоматизируемой предметной областью и программным кодом[5]. Такие технологии включают инструменты для быстрого пошагового преобразования моделей, выражающих разные точки зрения на задачи, в программы и структуры данных, с высокой степенью верифицируемости и трассируемости. К таким технологиям относятся инженерия предметной области (domain engineering), разработка, управляемая моделями (model-driven engineering, MDE), организация распределенных вычислений (distributed computing), аспектно-ориентированный подход (aspect-oriented software development, AOSD).

Применение таких технологий в проектировании больших систем требует масштабировать их – приспособить к гибкому манипулированию многочисленными, сложными, разнородными моделями[6]. Здесь необходима глубокая автоматизация, поэтому технологии жизненного цикла должны иметь единую формальную теоретическую базу, позволяющую кратко описать механизмы масштабирования, сформулировать и доказать их основные свойства, не «потонув» в деталях структуры частных моделей. Однако большинство формальных методов современной инженерии систем базируется на разнородных «тяжеловесных» математических средствах, подогнанных под разнообразные частные парадигмы программирования и вследствие этого плохо совместимых друг с другом. Технологии широкого назначения, подобные перечисленным выше, способные порождать рациональные типовые решения, развиваются в основном *ad hoc*, не опираясь на математические методы[7].

Таким образом, формирование теоретической основы технологий проектирования больших информационно-управляющих систем, свободной от указанных недостатков, является важной научной проблемой. В качестве математического аппарата, пригодного для ее решения, целесообразно привлечь теорию категорий. Эта теория позволяет явно и компактно выразить

---

[4] *Kannenberg A., Saiedian H.* Why software requirements traceability remains a challenge // J. Defense Software Engineering. July/August 2009. P. 14–19.

[5] *Morin B. et al.* Taming Dynamically Adaptive Systems using models and aspects // Proc. Intl. Conf. ICSE'2009. P. 122–132.

[6] *Kolovos D.S. et al.* The grand challenge of scalability for model driven engineering // Lecture Notes in Computer Sci. 2009. V. 5421. P. 48–53.

[7] *Diskin Z., Maibaum T.S.E.* Category theory and model-driven engineering: from formal semantics to design patterns and beyond // Proc. 7th Workshop ACCAT'2012. P. 1–21.

основные положения системной инженерии[8]. Артефактам[9] технологий можно сопоставить объекты подходящей категории (формальные модели), а технологическим процессам – морфизмы, перерабатывающие объекты-области (входы) в объекты-кообласти (выходы). Переходы между технологиями, сохраняющие структуру процессов, могут быть представлены функторами. Процедурам синтеза сложных моделей отвечают диаграммы в таких категориях («мегамодели»[10]), так что их анализ позволяет выявлять рациональные типовые решения, в том числе с привлечением автоматизированных инструментов[11, 12]. Для этого требуется построить теоретико-категорные конструкции, наиболее точно отражающие ключевые процедуры жизненного цикла, и доказать их основные свойства, важные с прикладной точки зрения. Общие конструкции такого рода редко встречаются в литературе – чаще изучаются частные категории, описывающие частные формальные методы.

Цель работы – повышение эффективности жизненного цикла больших информационно-управляющих систем путем создания единой формальной базы технологий их проектирования.

Достижение этой цели требует решения следующих **задач**:

- проведение системного анализа жизненного цикла больших информационно-управляющих систем;
- построение аппарата формального анализа и синтеза технологий проектирования систем на основе теории категорий;
- построение формальной технологии проектирования распределенных вычислительных систем;
- построение формальных технологий совместного аспектно-ориентированного моделирования данных и процессов;
- применение построенного формального аппарата для рационального проектирования прикладных информационно-управляющих систем.

Эти задачи решены в работе с использованием **методов** теории категорий, теории моделей, инженерии информационных систем.

Научная новизна работы состоит в том, что впервые построен и теоретически обоснован аппарат для математического (формального) анализа и

---

[8] *Goguen J.* Categorical foundations for general systems theory. In: Advances in Cybernetics and Systems Research. London: Transcripta Books, 1973. P. 121–130.

[9] Артефактом в инженерии программного обеспечения традиционно называется результат любой деятельности, выполняемой в рамках жизненного цикла, от лат. artefactum – искусственно сделанное (см. Большой энциклопедический словарь. М.: Изд-во «Большая Российская энциклопедия», 2000.)

[10] *Jouault F. et al.* Inter-DSL coordination support by combining megamodeling and model weaving // Proc. 2010 ACM Symp. on Applied Computing. P. 2011–2018.

[11] *Srinivas Y.V., Jüllig R.* SPECWARE: formal support for composing software // Lecture Notes in Computer Sci. 1995. Vol. 947. P. 399–422.

[12] *Khurshid N. et al.* Towards a tool support for specifying complex software systems by Categorical Modeling Language // Studies in Comp. Intelligence. 2010. Vol. 296. P. 133–149.

синтеза технологий проектирования программных систем на основе теории категорий, позволяющий находить рациональные типовые решения проблем масштабируемости (scalability), трассируемости (traceability), разделения ответственности (separation of concerns). Путем применения этого аппарата впервые построены математические (формальные) технологии, способные служить теоретической основой для широкого класса методов проектирования информационно-управляющих систем.

Практическая значимость работы заключается в применимости результатов для повышения эффективности жизненного цикла больших информационно-управляющих систем, в том числе для решения следующих практических задач:

- масштабирование технологий моделирования в целях обеспечения их применимости в жизненном цикле большой системы;
- отображение вычислительных алгоритмов на архитектуру гетерогенной распределенной вычислительной среды;
- трассирование артефактов жизненного цикла к классам задач, решаемых системой;
- расширение модульных технологий проектирования аспектно-ориентированными приемами и инструментами;
- совместное моделирование данных и процессов.

Положения, выносимые на защиту. На защиту выносятся:

- аппарат для математического (формального) анализа и синтеза технологий проектирования систем на основе теории категорий;
- алгебраические методы отображения алгоритмов на архитектуру распределенной вычислительной среды;
- теоретико-категорная семантика расширения модульных технологий проектирования систем аспектно-ориентированными приемами с обеспечением трассируемости;
- теоретико-категорные модели процедур идентификации, связывания и модуляризации аспектов;
- теоретико-категорные методы совместного моделирования данных и процессов.

Достоверность и обоснованность результатов. Корректность теоретических результатов, изложенных в диссертации, обоснована рядом теорем, снабженных подробными доказательствами. В подтверждение достоверности практических результатов автором получено 4 акта внедрения научных и практических результатов исследований, 5 свидетельств о регистрации программ для ЭВМ, 1 патент РФ на изобретение.

Внедрение результатов работы. При помощи подходов, предложенных в диссертации, рационально спроектированы и реализованы модели данных, процессов, вычислений и др. в следующих больших системах управления объектами топливно-энергетического комплекса:

- автоматизированная информационно-измерительная система коммерческого учета электроэнергии ООО «Транснефтьсервис-С» для ОАО «АК «Транснефть» (АИИС КУЭ ТНС, 2005-2007);
- автоматизированная информационно-измерительная система коммерческого учета электроэнергии ОАО «Томусинское энергоуправление» (АИИС КУЭ ТЭУ, 2008);
- автоматизированная система диспетчерского управления энергохозяйством ООО «Газпром энерго» (АСДУ ГПЭ, 2008-2009);
- единая интегрированная автоматизированная информационная система мониторинга и управления эффективностью энергосбережения на объектах города Москвы (ЕИАИС ЭЭ, 2010-2012).

Апробация работы. Результаты работы докладывались на следующих международных конференциях: 7th Joint European Networking Conference JENC7 (Budapest, Hungary, 1996); 8th Joint European Networking Conference JENC8 (Edinburg, Scotland, 1997); 3 Смирновские чтения (Москва, 2001); Международная конференция по вычислительной математике МКВМ-2004 (Новосибирск, 2004); Международная конференция «Алгебра, логика и кибернетика-2004» (Иркутск, 2004); Всероссийская научная конференция «Научный сервис в сети Интернет» (Новороссийск, 2004); IX рабочее совещание по электронным публикациям El-Pub2004 (Новосибирск, 2004); 2nd IASTED International Conference on Automation, Control and Information Technology ACIT-2005 (Новосибирск, 2005); Международная конференция «Диалог'2005» (Звенигород, 2005); XI International Conference “Knowledge-Dialogue-Solution” (Varna, Bulgaria, 2005); Международная конференция «Вычислительные и информационные технологии для наук об окружающей среде» CITES-2005 (Новосибирск, 2005); 9th Asian Logic Conference (Новосибирск, 2005); International Conference “Molecular spectroscopy and atmospheric radiative processes” (Томск, 2005); X Российская конференция с участием иностранных ученых «Распределенные информационно-вычислительные ресурсы» DICR-2005 (Новосибирск, 2005); II Международная научно-техническая конференция «Новые информационные технологии в нефтегазовой отрасли и образовании» (Тюмень, 2006); Международная конференция «Вычислительные и информационные технологии в науке, технике и образовании» (Павлодар, 2006); Международная конференция «Алгебра и ее приложения» (Красноярск, 2007); VII Международная научно-практическая конференция «Исследование, разработка и применение высоких технологий в промышленности» (Санкт-Петербург, 2009); 9th Workshop on Foundations of Aspect-Oriented Languages (Rennes, France, 2010); 3rd IASTED International Conference on Automation, Control and Information Technology ACIT-2010 (Новосибирск, 2010); XIII Российская конференция «Распределенные информационные и вычислительные ресурсы» DICR-2010 (Новосибирск, 2010); XI Международная научно-практическая конференция «Фундамен-

тальные и прикладные исследования, разработка и применение высоких технологий в промышленности» (Санкт-Петербург, 2011); Научная сессия НИЯУ МИФИ-2012 (Москва, 2012); XIV Международная конференция «Проблемы управления и моделирования в сложных системах» ПУМСС-2012 (Самара, 2012); VI Международная конференция «Управление развитием крупномасштабных систем» MLSD'2012 (Москва, 2012); Международная конференция «Алгебра и логика: теория и приложения» (Красноярск, 2013).

Также работа была представлена на научных семинарах Института проблем управления им. В.А. Трапезникова РАН, Вычислительного центра им. А.А. Дородницына РАН, Института вычислительных технологий СО РАН, Института математики СО РАН.

Публикации. По результатам выполненных исследований опубликовано 60 работ, в том числе: 17 в 10 ведущих рецензируемых изданиях, рекомендованных в действующем перечне ВАК для публикации результатов докторских диссертаций (из них 11 без соавторов), 1 учебное пособие, 1 патент, 5 свидетельств о регистрации программ для ЭВМ.

Личный вклад автора. Все теоретические результаты, вынесенные на защиту, получены автором лично и опубликованы в работах без соавторов. Работы, опубликованные в соавторстве, посвящены практической апробации результатов в ходе создания прикладных систем. В них автору принадлежат концептуальные модели и проектные решения.

Структура и объем работы. Диссертация состоит из введения, 5 глав, заключения, библиографии и приложения. Общий объем работы – 281 страниц, библиография содержит 264 наименований.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обосновывается актуальность темы диссертации, указываются цели и задачи исследования, определяются научная новизна и практическая значимость работы, приводится краткий обзор работы.

В **главе 1** изложены результаты системного анализа проблем снижения затрат на жизненный цикл больших информационно-управляющих систем. Показано, что основным источником больших значений масштабных факторов являются требования полноты (замкнутости) массивов сущностей, описывающих объект управления, относительно разнообразных отношений. Для регистрации и обработки пространственно-временных отношений успешно применяются технологии больших баз данных, параллельных и распределенных вычислений. Однако для отношений типа «цель–средство», порождающих многообразие функциональных задач системы, этого недостаточно, поскольку им присуща конфликтность, запутанность, изменчивость. Появляется большое количество рассеянных задач, не поддающихся локализации в рамках функционально замкнутых модулей с фиксированным интерфейсом.

Эффективное создание систем в условиях рассеяния является целью аспектно-ориентированного программирования (АОП)[13]. Реализация рассеянной задачи в АОП оформляется как аспект – особая программная единица, код которой автоматически вставляется в код других единиц в местах, явно задаваемых внешним образом. В результате вставки аспекты получают полный доступ к контексту, обходя ограничения модульного интерфейса.

Однако на практике АОП применяется значительно реже, чем модульные подходы, поскольку отсутствует единое непротиворечивое понимание его методологической основы[14]. На семантическом уровне неясно, как рационально выделять и комплексировать аспекты в системах и их моделях. Существующие технологии АОП предлагают только частные решения в рамках частных парадигм программирования. Они ограничены поддержкой программно-технических рассеянных задач, таких как журналирование, кэширование, защита информации и т.п. Не хватает типовых решений и инструментов для реализации семантически богатых функциональных рассеянных задач, таких как ведение паспорта объекта автоматизации, оповещение участников процессов о ходе их выполнения, оперативная оценка эффективности процессов, проверка правильности действий пользователей и компонентов системы, перевод информации на разные языки и в разные форматы. В связи с этим в работе предложена семантика АОП, основанная на концепции трассирования – прослеживания воплощения задач в артефактах жизненного цикла систем. Такой подход оправдан тем, что трассируемость страдает от рассеяния задач больше, чем другие показатели качества систем.

Типовые показатели качества больших информационно-управляющих систем систематизированы в форме модели, удовлетворяющей стандарту ISO/IEC 9126. Их допустимые значения выступают ограничениями в процессах жизненного цикла, в то время как многие критерии эффективности процессов состоят в минимизации затрат финансовых средств, труда и времени. Для больших систем в целях обеспечения возможности привлекать типовые технологии, способствующие снижению затрат, предложено организовать жизненный цикл на основе процесса инженерии предметной области (domain engineering, в ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств» – проектирование доменов). Его результатами являются модели и средства, предназначенные для многократного применения в непрерывных «прикладных» процессах создания и модификации компонентов и системы. Среди методов инженерии предметной области предпочтение от-

---

[13] *Kiczales G. et al.* Aspect-oriented programming // Lecture Notes in Computer Sci. 1997. V. 1241. P. 220–242.

[14] *Steimann F.* The paradoxical success of aspect-oriented programming // Proc. Intl. Conf. OOPSLA'2006. P. 481–497.

дается ориентированным на построение онтологий, таким как ODE (Ontology Domain Engineering)[15]. Онтология предметной области служит источником терминов и условий корректности для модели качества большой системы и детализирующих понятийных моделей (процессов, организационной структуры и т.д.). Для повышения эффективности процессов коллективной разработки онтологий в работе предложена инструментальная система ONTO-GRID, функционирующая в среде распределенных вычислений типа Grid.

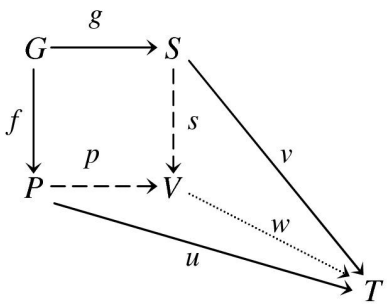
Чтобы создавать удобные предметно-ориентированные языки моделирования и автоматически порождать программы из моделей, следует применять технологии MDE, при условии их масштабирования. По мере роста масштаба системы возрастает трудоемкость согласования моделей между собой и с программным кодом, обеспечения корректности, производительности и др. При решении таких проблем целесообразно привлекать подходы, предложенные в работе. Строятся теоретико-категорные конструкции, формально описывающие решения на абстрактном концептуальном уровне. Путем вычислений в категориях оцениваются свойства решений, выбирается наиболее экономичное из альтернативных решений. Соответствующая ему абстрактная конструкция интерпретируется в понятиях подходящей технологии, и выбираются (либо создаются) инструменты для его реализации.

Глава 2 посвящена теоретико-категорному подходу к формализации процессов жизненного цикла. Здесь требуются категории, объектами которых служат формальные модели системных единиц, а морфизмами – действия по их интеграции. Обозначим категорию такого типа через *c-DESC*. Конфигурации взаимосвязанных компонентов, из которых собираются системы, описываются *c-DESC*-диаграммами – ориентированными графами, вершины которых помечены объектами, а ребра морфизмами категории *c-DESC*. Акту сборки системы отвечает копредел (colimit) диаграммы – коконус (cocone) над ней, обладающий свойством универсальности[16]. Например, рассмотрим комплексирование компонента P в систему S путем соединения – добавления промежуточного компонента G, называемого «клеем» (glue) или связкой (connector)[17], который способен интегрироваться как с компонентом, так и с системой. Путем соединения часто строятся системы на базе промежуточного ПО (middleware), которое и служит связкой. Конфигурация соединения имеет вид пары морфизмов $f : P \leftarrow G \rightarrow S : g$. Коконус над ней – это коммутативный квадрат: он задается объектом-вершиной V и парой морфизмов $p : P \rightarrow V \leftarrow S : s$, удовлетворяющей соотношению $p \circ f = s \circ g$. Коконус является копределом (и называется кодекартовым квадратом),

[15] *Falbo R.A. et al.* An ontological approach to domain engineering // Proc. Intl. Conf. SEKE'2002. P. 351–358.

[16] *Маклейн С.* Категории для работающего математика. М.: Физматлит, 2004.

[17] *Allen R. J., Garlan D.* A formal basis for architectural connection // ACM Transactions on Software Engineering and Methodology. 1997. V. 6(3). P. 213–249.



если он универсален в том смысле, что для любого коммутативного квадрата $u \circ f = v \circ g$ существует единственный морфизм w, удовлетворяющий соотношениям $w \circ p = u$ и $w \circ s = v$. Если такой кодекартов квадрат существует, то объект V действительно отвечает системе, которая собрана из S и P путем соединения посредством G (и не содержит ничего «лишнего»).

Более формально, диаграмма Δ в произвольной категории C – это функтор вида $\Delta : X \to C$, где X – малая категория, порожденная графом диаграммы и называемая ее схемой. Все диаграммы образуют категорию $\mathbf{D}C$ (ковариантная категория «сверхзапятой»[18]), в которой морфизмом диаграммы $\Delta : X \to C$ в $\Xi : Y \to C$ служит любая пара вида $\langle \varepsilon, fd \rangle$, состоящая из функтора $fd : X \to Y$ и естественного преобразования $\varepsilon : \Delta \to \Xi \circ fd$. Например, коконус над диаграммой Δ – это $\mathbf{D}C$-морфизм из Δ в одноточечную диаграмму, изображающую его вершину. Существует функтор вычисления схемы $sch : \mathbf{D}C \to \mathbf{CAT} : \Delta \mapsto \mathrm{dom}\,\Delta, \langle \varepsilon, fd \rangle \mapsto fd$. Любой функтор $fun : C \to D$ определяет функтор $fun \circ - : \mathbf{D}C \to \mathbf{D}D : \Delta \mapsto fun \circ \Delta, \langle \varepsilon, fd \rangle \mapsto \langle fun(\varepsilon), fd \rangle$, так что существует эндофунктор $\mathbf{D}$ в категории $\mathbf{CAT}$ всех категорий и функторов.

Для более полного описания процесса синтеза систем формализуются понятия интеграционного интерфейса и трансформации. Например, у вебсервиса интерфейсом служит его описание на языке WSDL, а трансформацией – реализация его спецификации на языке программирования. Формальные модели интерфейсов образуют категорию, обозначаемую через *SIG*, а выделению интерфейсов отвечает функтор $sig : c\text{-}DESC \to SIG$. Интерфейсы должны в полном объеме описывать интеграционные возможности компонентов, а также быть реализуемыми: должен существовать функтор дискретной реализации $sig^* : SIG \to c\text{-}DESC$ такой, что $sig \circ sig^* = 1_{SIG}$ и существует биекция $sig : \mathrm{Mor}(sig^*(I), S) \cong \mathrm{Mor}(I, sig(S))$ для любых $I \in \mathrm{Ob}\, SIG$, $S \in \mathrm{Ob}\, c\text{-}DESC$ (это свойство компактно формулируется в терминах сопряжения функторов). Трансформациям отвечают морфизмы категории, обозначаемой через *r-DESC*, объектами которой являются формальные модели. Накладываются условия естественности выделения интерфейса и трансформации относительно сборки систем. В частности, покомпонентная трансформация конфигурации Δ формально представляет собой семейство *r-DESC*-морфизмов, образующее естественное преобразование дискретной диаграммы $|\Delta|$, полученной из Δ путем удаления всех нетождественных морфизмов.

В итоге получается следующая сложная конструкция[19]:

---

[18] *Маклейн С.* Категории для работающего математика. М.: Физматлит, 2004.

[19] *Fiadeiro J.L. et al.* A mathematical semantics for architectural connectors // Lecture Notes in Computer Sci. 2003. V. 2793. P. 190–234.

Определение 2.3. Пусть задана категория *c-DESC*, объектами которой служат формальные модели компонентов и систем, а морфизмами – действия по интеграции (компонентов в системы). *Формальной технологией проектирования* (architecture school) над *c-DESC* называется четверка $\langle c\text{-}DESC, Conf, sig, r\text{-}DESC\rangle$, где:

- *Conf* – класс *c-DESC*-диаграмм, называемых допустимыми конфигурациями систем (configurations);
- *sig* – функтор из *c-DESC* в категорию, обозначаемую *SIG*, объекты которой называются интерфейсами или сигнатурами компонентов (signatures), а морфизмы – действиями по интеграции интерфейсов;
- *r-DESC* – категория, объектами которой являются формальные модели, а морфизмы называются трансформациями компонентов.

При этом выполняются следующие условия.

(i) Любая диаграмма из класса *Conf* имеет копредел.

(ii) Функтор *sig* унивалентен.

(iii) Функтор *sig* обладает левым сопряженным, обозначаемым через *sig**, с тождественной единицей сопряжения.

(iv) Функтор *sig* поднимает копределы всех диаграмм из класса *Conf*.

(v) Для любых *c-DESC*-диаграмм Δ, Ξ, если $sig \circ \Delta = sig \circ \Xi$ и $\Delta \in Conf$, то $\Xi \in Conf$.

(vi) Ob *r-DESC* = Ob *c-DESC*.

(vii) Подкатегория в *c-DESC*, состоящая из всех *c-DESC*-объектов и всех изоморфизмов, является подкатегорией в *r-DESC*.

(viii) Для любых диаграммы $\Delta \in Conf$ и естественного преобразования вида $\varphi : |\Delta| \to \Sigma \in \mathrm{Mor}\ r\text{-}DESC^{|sch(\Delta)|}$ существуют диаграмма $\Delta \oplus \varphi \in Conf$, содержащая Σ в качестве поддиаграммы, и *r-DESC*-морфизм $r : colim(\Delta) \to colim(\Delta \oplus \varphi)$. □

Тройка $\langle c\text{-}DESC, Conf, sig\rangle$, удовлетворяющая условиям (i)-(v), называется *формальной технологией специфицирования*, а пара $\langle c\text{-}DESC, Conf\rangle$, удовлетворяющая условию (i) – *формальной технологией конфигурирования*.

В качестве примера в работе построена формальная технология моделирования сценариев выполнения процессов $SM = \langle \mathbf{Pos}, CPos, |–|, r\text{-}Pos\rangle$, где:

- **Pos** – категория всех частично упорядоченных множеств и всех их монотонных отображений (сценарий задается множеством событий, частично упорядоченных причинно-следственными связями);
- *CPos* – класс всех копроизведений **Pos**-коконусов (результат сборки сценариев задается явно, с точностью до раздельного объединения);
- $|–| : \mathbf{Pos} \to \mathbf{Set}$ – канонический функтор, забывающий порядок (переход к множеству событий представляет собой единственный, с точностью до эквивалентности категорий, нетривиальный способ выделения интерфейса у сценария);

- *r-Pos* – категория, объектами которой являются все частично упорядоченные множества, а морфизмом X в Y служит любое антифункциональное тотальное отношение $R \subseteq X \times Y$ такое, что $\forall x, x' \in X\ \forall y, y' \in Y$ $((xRy \wedge x'Ry' \wedge x \neq x') \Rightarrow (x \leq x' \Leftrightarrow y \leq y'))$ (трансформация сценария заключается в расширении событий до подсценариев с полным наследованием порядка).

В работе доказано, что в любой формальной технологии функтор *sig детерминирует* (т.е. не только поднимает, но и сохраняет) копределы всех конфигураций. Для анализа их состава введена следующая конструкция:

Определение 2.6. *SIG*-диаграмма Θ называется (*sig*-)*предконфигурацией*, если она имеет копредел и функтор *sig* поднимает копределы всех диаграмм из класса $\mathbf{D}sig^{-1}(\{\Theta\}) = \{\Delta \mid sig \circ \Delta = \Theta\}$. Класс всех *c-DESC*-диаграмм, переходящих в предконфигурации под действием функтора **D***sig*, называется *потенциалом комплексируемости* функтора *sig*. □

Показано, что в любой формальной технологии класс *Conf* имеет вид $\mathbf{D}sig^{-1}(IC)$ для некоторого класса *sig*-предконфигураций *IC*. Поэтому для заданного функтора *sig* важно знать, насколько богат его потенциал комплексируемости. Его наименьшим возможным значением является пустое множество: существуют непустая категория *c-DESC* и функтор *sig* такие, что тройка $\langle c\text{-}DESC, Conf, sig\rangle$ является технологией специфицирования тогда и только тогда, когда класс *Conf* пуст. А наиболее богатый потенциал комплексируемости – это класс всех *c-DESC*-диаграмм, имеющих копределы: таким потенциалом обладают функторы, которые поднимают копределы всех диаграмм. Технология, в которой интерфейсы выделяются таким функтором, называется *скоординированной*[20]. В работе доказано, что формальная технология скоординирована тогда и только тогда, когда имеет место разложение $sig = sig' \circ se$, где sig' – некоторый топологический функтор, *se* – эквивалентность категорий, сюръективная на объектах. Примером служит технология специфицирования систем множествами аксиом – предложений формального исчисления фиксированной сигнатуры, выступающей в качестве единственного интерфейса. Пусть σ – сигнатура, $\mathrm{L}(\sigma)$ – язык (множество всех правильно построенных предложений) сигнатуры σ, $\vdash$ – отношение выводимости, $LS_\sigma = (2^{\mathrm{L}(\sigma)}, \{(S, T) \mid S, T \subseteq \mathrm{L}(\sigma), T \vdash S\})$ – предупорядоченное множество всех спецификаций сигнатуры σ, $ddLS_\sigma$ – класс всех дискретных LS_σ-диаграмм. Четверка

$$TS_\sigma = \langle LS_\sigma, ddLS_\sigma, \sigma(-) : LS_\sigma \to (\{\sigma\}, \{1_\sigma\}), (\mathrm{Ob}\ LS_\sigma, \mathrm{Iso}\ LS_\sigma)\rangle$$

является скоординированной формальной технологией проектирования.

Интерфейсы играют важную роль при распараллеливании – одном из основных механизмов масштабирования, который состоит в разбиении систе-

[20] *Fiadeiro J.L.* Categories for Software Engineering. Berlin Heidelberg N. Y.: Springer, 2005.

мы на части, в той или иной степени изолированные друг от друга. В работе предложено формально оценивать степень изоляции компонентов как степень структурного сходства модели системы с копроизведением моделей своих компонентов. Отличие от копроизведения возможно ввиду взаимовлияния компонентов (синергии), однако на уровне интерфейсов это сходство должно иметь вид изоморфизма, поскольку иначе компоненты нельзя считать образующими разбиение системы. Кроме того, в распараллеливании не должны участвовать «лишние» компоненты, не вносящие никакого вклада в интерфейс системы. Формально, *разбиением c-DESC*-объекта *S* назван коконус π с вершиной *S*, удовлетворяющий следующим условиям:

(i) Основание коконуса π является дискретной диаграммой.
(ii) π является подкоконусом копредела некоторой конфигурации.
(iii) $sig \circ \pi$ является копределом (т.е. копроизведением).
(iv) Если θ – собственный подкоконус коконуса π, то $sig \circ \theta$ не является копределом.

Разнообразные методы распараллеливания порождают различные классы разбиений. Например, разбиение называется *суммой*, если оно само является копределом, так что задает максимальную степень взаимной изоляции компонентов. В работе классы разбиений погружаются в **D***c-DESC* в качестве подкатегорий, в которых удается формально вычислять разбиения заданного *c-DESC*-объекта, обладающие экстремальным характером: им отвечают универсальные объекты – инициальные (наиболее «мелкие» разбиения) и терминальные (наиболее «крупные»).

Пусть π – разбиение в технологии моделирования сценариев *SM*. Основные множества всех объектов его основания образуют разбиение (в буквальном смысле) основного множества сценария *S*. В частности, сопоставление сценарию множества всех его связных компонент (взаимно независимых подсценариев), образующих наиболее «мелкую» сумму, порождает функтор из **Pos** в **Set**, сопряженный слева к функтору дискретной реализации интерфейса сценариев, причем коединица этого сопряжения тождественна. Технологии, обладающие функтором такого рода, названы в работе *структурируемыми*: они предоставляют широкие возможности для распараллеливания.

Существуют формальные технологии проектирования, в которых *r-DESC* является подкатегорией в *c-DESC*. В них покомпонентную трансформацию конфигурации Δ семейством $\varphi \in \mathrm{Mor}\, r\text{-}DESC^{|sch(\Delta)|}$ можно изобразить *натяжкой* (pull) – *c-DESC*-диаграммой $\Delta \rightrightarrows \varphi$, состоящей из диаграммы Δ и стрелок, исходящих из всех вершин ее графа и помеченных соответствующими компонентами семейства φ. Если натяжку можно использовать в качестве диаграммы $\Delta \oplus \varphi$ из условия (viii) определения 2.3, то такая технология называется (трансформационно) *однородной*. Однородные технологии возникают в алгебраическом подходе к проектированию, где и интеграция, и трансформация сводятся к обогащению вычислительных возможностей мо-

дели, описываемых множеством всех термов некоторой алгебры (типа данных). Пример такой технологии построен в работе (гл.3 ниже).

Также рассмотрены формальные технологии, двойственные к однородным: в них *r-DESC* является подкатегорией в $c\text{-}DESC^{\mathrm{op}}$. Здесь появляется возможность трассирования: трасса получается из трансформации путем обращения ее направления. Покомпонентная трансформация конфигурации Δ изображается в виде ее *накачки* (push) семейством трасс τ – *c-DESC*-диаграммы $\tau \rightrightarrows \Delta$, состоящей из диаграммы Δ и стрелок, входящих в вершины ее графа и помеченных компонентами семейства τ. Накачка имеет копредел с тем же объектом, что и Δ. Поэтому трансформации, двойственные к действиям по интеграции, являются неразрушающими (non-invasive) по отношению к сборке систем – если накачка является конфигурацией, то она используется в качестве $\Delta \oplus \tau^{\mathrm{op}}$. Технологии такого рода называются *координородными*. К ним относятся технология моделирования сценариев *SM*, а также технологии моделирования предметных областей графами (такими как онтологии, схемы процессов и т.д.), где трансформации состоят в углублении предметных знаний, приводящем к замене вершин подграфами.

Сложные технологии создания систем можно комплексировать из простых так же, как сами системы. Такой подход характерен для MDE, и в целях его теоретического обоснования в работе построена формальная технология проектирования, моделями в которой служат технологии конфигурирования. Действием по интеграции $cm : \langle c\text{-}DESC_1, Conf_1\rangle \to \langle c\text{-}DESC_2, Conf_2\rangle$ служит любой функтор $cm : c\text{-}DESC_1 \to c\text{-}DESC_2$, сохраняющий конфигурации и естественный (в традиционном «слабом» смысле) относительно комплексирования систем. Все формальные технологии конфигурирования и все действия по их интеграции (со стандартным законом композиции функторов) образуют категорию **CONF**. Интерфейсы извлекаются из них посредством функтора $\boldsymbol{desc} : \mathbf{CONF} \to \mathbf{CAT} : \langle c\text{-}DESC, Conf\rangle \mapsto c\text{-}DESC$, «забывающего» конфигурации. Его потенциал комплексируемости *DSConf* определяет возможности комплексирования технологий. Трансформации технологий отражают реализацию систем, т.е. переход от технологий конфигурирования интерфейсов к технологиям конфигурирования их реализаций. Их трассы образуют класс *r-CONF* всех **CONF**-морфизмов вида $sig : \langle c\text{-}DESC, Conf\rangle \to \langle SIG, sig \circ Conf\rangle$ таких, что функтор *sig* удовлетворяет условиям (ii)-(v) определения 2.3. *Технология синтеза технологий конфигурирования* имеет вид

$$\boldsymbol{SCONF} = \langle \mathbf{CONF}, DSConf, \boldsymbol{desc}, (\mathrm{Ob}\ \mathbf{CONF}, r\text{-}CONF)^{\mathrm{op}}\rangle.$$

Аналогичным образом в работе строятся технологии синтеза технологий специфицирования и проектирования:

Определение 2.13. *Морфизмом формальной технологии проектирования* $\langle c\text{-}DESC_1, Conf_1, sig_1 : c\text{-}DESC_1 \to SIG_1, r\text{-}DESC_1\rangle$ в технологию

$\langle c\text{-}DESC_2, Conf_2, sig_2 : c\text{-}DESC_2 \to SIG_2, r\text{-}DESC_2\rangle$ называется тройка функторов

$$\langle cm : c\text{-}DESC_1 \to c\text{-}DESC_2, sm : SIG_1 \to SIG_2, rm : r\text{-}DESC_1 \to r\text{-}DESC_2\rangle,$$

удовлетворяющая следующим условиям:

(i) $cm \circ Conf_1 \subseteq Conf_2$.

(ii) cm сохраняет копределы всех диаграмм из $Conf_1$.

(iii) $sig_2 \circ cm = sm \circ sig_1$.

(iv) $rm(i) = cm(i)$ для любого $i \in$ Iso $c\text{-}DESC_1$.

Пара функторов $\langle cm, sm\rangle$, удовлетворяющая условиям (i)-(iii), называется *морфизмом формальной технологии специфицирования* $\langle c\text{-}DESC_1, Conf_1, sig_1\rangle$ в $\langle c\text{-}DESC_2, Conf_2, sig_2\rangle$. Функтор cm, удовлетворяющий условию условиям (i)-(ii), называется *морфизмом формальной технологии конфигурирования* $\langle c\text{-}DESC_1, Conf_1\rangle$ в $\langle c\text{-}DESC_2, Conf_2\rangle$. Если все компоненты морфизма технологий являются вложениями подкатегорий, то область морфизма называется *подтехнологией* его кообласти. □

При помощи этих конструкций вводятся категории технологий **ARCH** и **SPEC** по аналогии с **CONF**. Существуют очевидные «забывающие» функторы ***conf*** : **SPEC** → **CONF**, ***spec*** : **ARCH** → **SPEC**.

В **главе 3** построена и теоретически обоснована формальная технология проектирования вычислительных систем. Для достижения эффективности функции расчета и анализа данных в составе больших систем реализуются в среде распределенных вычислений с динамическим развертыванием типа Grid или облака, узлы которой обладают разными ограничениями по объему доступных ресурсов памяти и поддерживают разнообразные модели вычислений. В то же время в спецификациях вычислительных задач алгоритмы описываются в терминах абстрактных типов данных (числа, списки и т.д.), образующих бесконечные множества и не предполагающих распределенного доступа. Поэтому в ходе проектирования алгоритмов необходимо выполнять их отображение на архитектуру вычислительной среды[21] – редукцию на конечные множества доступных ресурсов, подбор средств управления потоком вычислений, комплексирование вычислительных компонентов в распределенные системы, оценку и минимизацию сложности. В контексте MDE отображение на архитектуру является частным случаем перехода от платформенно-независимых моделей (PIM) к платформенно-зависимым (PSM)[22].

Такой переход может быть сопряжен со значительными затратами труда, поэтому в целях его автоматизации в работе предложена алгебраическая формализация отображения. Основным понятием является модель вычисле-

[21] *Воеводин В.В.* Отображение проблем вычислительной математики на архитектуру вычислительных систем // Вычисл. методы и программирование. 2000. Т.1. С. 37–44.

[22] *Frankel D.S.* Model Driven Architecture: Applying MDA to Enterprise Computing. N. Y.: Wiley & Sons, 2003.

ний – конечная алгебра, состоящая из совокупности кодов поддерживаемых чисел и вычислительных примитивов, отвечающих аппаратным инструкциям, операторам языка программирования или библиотечным функциям математического пакета программ. Все вычислительные операции модели образуют клон этой алгебры. Действия по интеграции моделей вычислений формализуют перекодирование поддерживаемых чисел при взаимодействии, так что интерфейсом модели служит ее основное множество. Трансформациями моделей вычислений являются в точности все модификации сигнатур алгебр и их интерпретаций, обогащающие (либо сохраняющие) их клоны. Так строится технология проектирования вычислительных систем.

В рамках этой технологии проводится анализ сложности компьютерных реализаций алгоритмов – результатов их отображения. Сложность выполнения действия в модели вычислений оценивается как минимальное необходимое количество обращений к ее примитивам[23]. В оценку сложности вычислений в распределенной среде входит также минимальное количество актов взаимодействия компонентов. Таким образом, потоки вычисления алгоритмов задаются комбинациями термов и морфизмов – абстрактными инвариантами их поведения, для которых разработаны общие методы оценки производительности[24]. Инварианты поведения пригодны к прямой автоматической трансляции в программный код с обеспечением корректности, что позволяет реализовать подход MDE к разработке вычислительных систем.

Для формального синтеза моделей вычислений с учетом ресурсных ограничений в работе предложен метод частичной интерпретации теории первого порядка T. Он состоит в построении модели для совокупности всех формул, выводимых из T, вычисление которых не выходит за рамки конечного набора сигнатурных констант. В качестве T берутся арифметики на тех или иных классах чисел, обогащенные достаточным количеством числовых констант. Их частичными интерпретациями выступают распространенные машинные реализации арифметики. Они обогащаются средствами управления потоком вычислений, в частности функциями $\mathrm{If}^a(x, y, z)$, отвечающими условным операторам `if x = a then y else z`. Поддержка таких функций характеризует PAM-машину – классическую модель вычислительного устройства[25]. Функции If^a являются термальными операциями конечной алгебры $\mathfrak{A}$ для всех $a \in |\mathfrak{A}|$ тогда и только тогда, когда ее клон $\mathrm{Clo}\,\mathfrak{A}$ состоит из всех функций, сохраняющих все ее подалгебры. Такие алгебры называются полупримальными[26].

---

[23] Дискретная математика и математические вопросы кибернетики. Т. I. М.: Наука, 1974.

[24] *Смелянский Р.Л.* Методы анализа и оценки производительности вычислительных систем. М.: МГУ, 1990.

[25] *Ахо А.В. и др.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

[26] *Foster A.L., Pixley A.F.* Semi-categorical algebras I. Semi-primal algebras // Mathematische Zeitschrift. 1964. Vol. 83, N 2. P. 147–169.

Для моделирования процессов синтеза вычислительных систем традиционное понятие гомоморфизма алгебр является слишком узким. Интеграция вычислительного компонента в систему состоит в кодировании значений чисел, поддерживаемых компонентом – в сопоставлении им значений, поддерживаемых системой. Формально кодированию отвечает отображение основных множеств алгебр, не разрушающее структуру тех вычислительных операций, которые компонент должен выполнять в составе системы. Это условие формулируется очевидным образом при взаимно-однозначном кодировании. Будем говорить, что биекция множеств $\iota : A \leftrightarrow B$ индуцирует вложение некоторого клона X_A в клон Y_B, если для любого $k > 0$ и любой k-местной функции $t \in X_A$ функция $\iota(t) : (x_1, \ldots, x_k) \mapsto \iota t(\iota^{-1}x_1, \ldots, \iota^{-1}x_k)$ принадлежит Y_B. Это означает, что вычислительные возможности клона X_A не превосходят возможностей клона Y_B. При помощи этой конструкции в работе вводится обобщение гомоморфизмов алгебр:

Определение 3.5. Пусть $\mathfrak{A}$ и $\mathfrak{B}$ – конечные алгебры. Отображение $\varphi : |\mathfrak{A}| \to |\mathfrak{B}|$ называется *структурным*, если биекция множеств $\mathrm{id}\,\varphi : |\mathfrak{A}|/\ker\varphi \leftrightarrow \varphi(|\mathfrak{A}|)$ индуцирует вложение клона $\mathrm{Clo}\,\mathfrak{A}/\varphi = (\mathrm{Clo}\,\mathfrak{A} \bigcap \mathrm{Pol}\,\{\ker\varphi\})/\ker\varphi$ в клон $\mathrm{Clo}\,\mathfrak{B}\restriction_\varphi = (\mathrm{Clo}\,\mathfrak{B} \bigcap \mathrm{Pol}\,\{\varphi(|\mathfrak{A}|)\})\restriction_{\varphi(|\mathfrak{A}|)}$. □

Здесь через $\mathrm{Pol}\,\{R\}$ обозначена совокупность всех функций, сохраняющих отношение R. Для моделирования вычислительных систем нужны категории, объектами которых служат все конечные алгебры, а морфизмами – структурные отображения, в том числе гомоморфизмы, и кроме того, инъекции – включения подсистем без «склеивания» кодов. Будем называть такие категории *структурными*. Среди них нет наибольшей, которая могла бы формально задать все возможные системные взаимосвязи конечных алгебр. Тем не менее, в работе построены структурные категории, предоставляющие наиболее богатые средства формализации процессов комплексирования вычислительных систем. В таких категориях можно оценивать интеграционные возможности моделей вычислений. Такие категории выделяются тем, что они должны содержать в качестве морфизмов как можно больше структурных отображений полупримальных алгебр. Кроме того, должна быть возможность представить процедуру превращения произвольной конечной алгебры в вычислительный компонент как действие рефлектора – функтора, полностью воспроизводящего ее возможности по интеграции в системы:

Определение 3.7. *Подкатегорией вычислительных систем* произвольной структурной категории алгебр *SCA* (обозначается $\mathrm{cs}(SCA)$) называется полная подкатегория в *SCA*, класс объектов которой состоит из всех полупримальных алгебр. Категория *SCA* называется *нормальной*, если $\mathrm{cs}(SCA)$ рефлективна в *SCA*. □

Теорема 3.1. Существует нормальная структурная категория, содержащая подкатегорию вычислительных систем любой структурной категории. □

В подкатегории вычислительных систем категории, удовлетворяющей условию теоремы 3.1, существует слишком «мало» (ко)пределов (в ней в общем случае не определены ни произведения пар, ни коуравнители). Формальная технология проектирования вычислительных систем, основанная на ней, не обладала бы свойством скоординированности. Однако это свойство существенно для формализации динамического развертывания, в условиях которого любая конечная конфигурация вычислительных компонентов должна комплексироваться в систему. Поэтому более подходящей основой для формализации проектирования вычислительных систем служит наименьшая нормальная категория, обозначаемая в работе через CN:

Теорема 3.2. Подкатегория вычислительных систем нормальной структурной категории эквивалентна некоторой топологической категории над **FinSet** тогда и только тогда, когда она совпадает с cs(CN). □

Отображение основных множеств алгебр названо в работе *субнепрерывным*, если прообраз любой подалгебры относительно него является подалгеброй. Положим CT = cs(CN). Класс Mor CT совпадает с классом всех субнепрерывных отображений полупримальных алгебр. Категория CT конечно кополна (а также конечно полна), и четверка

$$\mathrm{CSD} = \langle \mathrm{CT},\ \mathrm{Ob}\,(\mathbf{FinCat} \downarrow \mathrm{CT}),\ |\text{–}| : \mathrm{CT} \to \mathbf{FinSet},\ (\mathrm{Ob\ CT},\ \mathrm{Epi\ CT} \cap \mathrm{Mono\ CT})\rangle$$

является скоординированной и, кроме того, структурируемой однородной формальной технологией проектирования.

Рефлектор cs : CN → CT формализует построение моделей вычислений из произвольных конечных алгебр, однако не в полной мере отражает процедуру получения практически полезных вычислительных систем. В целях согласования вычислительных компонентов по данным выделяют структурные отображения, сохраняющие все константы: такие CN-морфизмы названы в работе *главными*. Кроме того, выделяют *флаг* – специальную константу, возвращаемую при переполнениях и округлениях. Он позволяет задавать частичные структурные отображения, отвечающие неполному кодированию данных: незакодированным числам сопоставляется флаг. Пусть $*$ – флаг, CN_* – подкатегория в CN, состоящая из всех конечных алгебр и всех главных морфизмов, сохраняющих константу $*$, CS_* – полная подкатегория в CN_*, объектами которой являются все полупримальные алгебры, в которых имеется константа $*$. CS_* рефлективна в CN_* и эквивалентна топологической категории над категорией **PFinSet** всех конечных множеств и всех их частичных отображений. Получается скоординированная структурируемая однородная формальная технология

$$\mathrm{CSD}_* = \langle \mathrm{CS}_*,\ \mathrm{Ob}\,(\mathbf{FinCat} \downarrow \mathrm{CS}_*),\ |\text{–}| \setminus \{*\} : \mathrm{CS}_* \to \mathbf{PFinSet},\ (\mathrm{Ob\ CS}_*,\ \mathrm{Epi\ CS}_* \cap \mathrm{Mono\ CS}_*)\rangle.$$

Определение 3.10. *Вычислительным расширением* конечной алгебры $\mathfrak{A}$ с флагом $*$ называется алгебра $cs_*(\mathfrak{A})$, где $cs_* : CN_* \to CS_*$ – рефлектор. □

Для произвольной сигнатуры σ в работе построен функтор из категории теорий LS_σ в CS_*, переводящий любую непротиворечивую теорию сигнатуры σ в вычислительное расширение ее частичной интерпретации. Однако этот функтор не индуцирует морфизм технологии аксиоматизации спецификаций TS_σ в технологию проектирования вычислительных систем CSD_* даже на уровне технологий конфигурирования. Это свидетельствует о концептуальном разрыве между постановкой и реализацией вычислительных задач.

Примером вычислительного расширения служит широко используемая модель целочисленных вычислений по mod n, $n > 1$, с операциями из кольца $\mathbf{Z}/n\mathbf{Z}$[27]. Для ее анализа в работе применяется аппарат конечнозначной логики Лукасевича. Логической матрицей Лукасевича[28] называется алгебраическая система $Ł_{n+1} = \langle E_{n+1}, \sim, \to, D_n \rangle$, где $\sim x = n - x$, $x \to y = \min(n, n - x + y)$, $D_n(x) \Leftrightarrow (x = n)$. Тем же символом $Ł_{n+1}$ обозначается алгебра, получающаяся из матрицы Лукасевича путем обеднения до сигнатуры $\{\sim, \to\}$.

Теорема 3.3. Вычислительное расширение кольца $\mathbf{Z}/n\mathbf{Z}$ с флагом n изоморфно $Ł_{n+1}$. □

Этот результат позволяет привлечь технику доказательства логики Лукасевича (в том числе автоматизированного[29]) для анализа корректности и сложности компьютерных реализаций вычислительных алгоритмов. В связи с этим в работе построен базис в $Ł_{n+1}$, содержащий арифметические операции по модулю n и условный оператор обработки флага If^n.

В **главе 4** построена и теоретически обоснована универсальная теоретико-категорная семантика расширения технологий модульного проектирования систем приемами аспектно-ориентированного подхода. Она позволяет дополнить произвольную формальную технологию проектирования $AR = \langle c\text{-}DESC, Conf, sig, r\text{-}DESC \rangle$ конструкциями, описывающими трассирование, связывание, модуляризацию аспектов. Трассированию подлежат в первую очередь трансформации, поскольку они меняют природу артефактов. Как указывалось выше, трасса r-*DESC*-морфизма $r : S \to T$, если она существует, задается c-*DESC*-морфизмом $r^{op} : T \to S$. Совместно трассировать трансформацию и сборку систем можно непосредственно, если трасса r^{op} обратима справа. Действительно, существование c-*DESC*-морфизма $s : S \to T$ такого, что $r^{op} \circ s = 1_S$, эквивалентно существованию для любого c-*DESC*-морфизма $p : X \to S$ действия по интеграции X в T, совместимого с трассиро-

[27] Здесь $\mathbf{Z}/n\mathbf{Z}$ рассматривается как кольцо без единицы (точнее, как обеднение кольца вычетов до сигнатуры кольца без единицы).

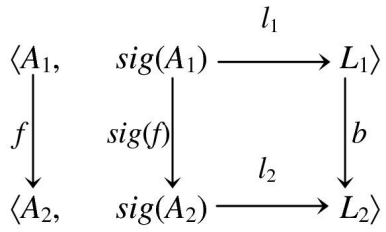
[28] *Карпенко А.С.* Логики Лукасевича и простые числа. М.: Наука, 2000.

[29] *Beavers G.* Automated theorem proving for Łukasiewicz logics // Studia Logica. 1993. Vol. 52, No. 2. P. 183–195.

ванием трансформации r в том смысле, что композиция трассы r^{op} с этим действием дает p (таким действием служит $s \circ p$, поскольку $r^{op} \circ (s \circ p) = p$).

Морфизм s, будучи обратимым *слева*, является регулярным мономорфизмом – теоретико-категорным аналогом включения источника трансформации в результат, не разрушающего структуру. На практике его построение может быть трудоемким, но он требуется не всегда, поскольку трассированию вдоль процессов интеграции систем подлежат в первую очередь интеграционные требования, предъявляемые к интерфейсам моделей. Следуя подходу к трассируемости, основанному на значимости (value-based requirements traceability)[30], в работе обратимость справа требуется от *SIG*-морфизма $sig(r^{op})$, который назван *sig-разметкой*. Реализация его обращения обычно не требует значительных затрат, поскольку интерфейсы проектируются так, чтобы интегрировать их было «проще», чем сами модели.

Семантика трассирования подсказывает способ расширения модульных технологий проектирования, приводящий к достижению цели аспектно-ориентированного подхода: нужно обогащать модели трассируемыми трансформациями, порождающими их из классов задач. Это позволяет синхронизировать изменения моделей с изменениями задач. Поскольку для АОП существенно в первую очередь влияние трансформаций на интеграционные возможности моделей, достаточно присоединить к моделям действия трансформаций на уровне интерфейсов, т.е. разметки. Для этого в работе привлекается конструкция категории запятой[31], а именно категория $sig \downarrow SIG$. Ее



объектами являются все пары вида $\langle A, l : sig(A) \to L\rangle$, где $A \in \mathrm{Ob}\ c\text{-}DESC$ и $l \in \mathrm{Mor}\ SIG$. Морфизмом объекта $\langle A_1, l_1 : sig(A_1) \to L_1\rangle$ в $\langle A_2, l_2 : sig(A_2) \to L_2\rangle$ является любая пара $\langle f : A_1 \to A_2,\ b : L_1 \to L_2\rangle$ такая, что $b \circ l_1 = l_2 \circ sig(f)$.

В работе используются $(sig \downarrow SIG)$-объекты специального вида:

Определение 4.3. *Аспектно-ориентированной моделью* (АО-моделью) называется любой $(sig \downarrow SIG)$-объект $\langle A, l : sig(A) \to L\rangle$ такой, что l является sig-разметкой. c-*DESC*-объект A называется (*модульной*) *основой* АО-модели, *SIG*-морфизм l – ее (*аспектной*) *разметкой*, *SIG*-объект L – ее *аспектной структурой*. □

Обозначим через *AO* полную подкатегорию в $sig \downarrow SIG$, класс объектов которой состоит из всех АО-моделей. Например, в технологии *SM* разметка – это сюръективное отображение, сопоставляющее каждому событию обозначение задачи, при выполнении которой оно возникло. Такой подход к моде-

[30] *Egyed A. et al.* Value-based requirements traceability: lessons learned // Lecture Notes in Business Information Processing. 2009. Vol. 14. P. 240–257.
[31] *Маклейн С.* Категории для работающего математика. М.: Физматлит, 2004.

лированию сценариев был предложен еще в 1980-х годах[32], однако природа меток и способы их синтеза оставались неясными, поскольку они не рассматривались в контексте АОП. Моделирование поведения систем помеченными сценариями выступает в роли операционной семантики АОП: его упрощенный частный случай известен как трассовая семантика аспектов[33]. Средства инженерии процессов оперируют с разнообразными классами помеченных сценариев, позволяя записывать их в специализированных нотациях: графических, алгебраических, гипертекстовых, сетей Петри и др.

Через SIG^{2} обозначается категория стрелок[34], класс объектов которой состоит из всех *SIG*-морфизмов, а морфизмом из f в g является любая пара *SIG*-морфизмов $\langle u, v\rangle$ такая, что $v \circ f = g \circ u$. Определены функторы $dom, codom : SIG^{2} \to SIG$, переводящие любой *SIG*-морфизм в его область и кообласть, соответственно. Пусть *LAB* – полная подкатегория в SIG^{2}, класс объектов которой состоит из всех *sig*-разметок, $il : LAB \hookrightarrow SIG^{2}$ – вложение. Следующие «забывающие» функторы связаны с категорией $sig \downarrow SIG$:

- $mod : AO \to c\text{-}DESC : \langle A, l\rangle \mapsto A, \langle f, b\rangle \mapsto f$;
- $asp : AO \to LAB : \langle A, l\rangle \mapsto l, \langle f, b\rangle \mapsto \langle sig(f), b\rangle$;
- $int = sig \circ mod = dom \circ il \circ asp : AO \to SIG : \langle A, l\rangle \mapsto sig(A), \langle f, b\rangle \mapsto sig(f)$;
- $str = codom \circ il \circ asp : AO \to SIG : \langle A, l\rangle \mapsto \mathrm{codom}\, l, \langle f, b\rangle \mapsto b$.

В работе функторы 1_{AO}, *mod*, *asp* и *int* названы *фундаментальными*, поскольку они позволяют извлекать из АО-моделей различные интеграционные интерфейсы в смысле определения 2.3 (см. теорему 4.1 ниже). Модульный интерфейс (*mod*) играет ключевую роль при модуляризации аспектов, аспектный (*asp*) – при синтезе разметок, исходный (*int*) – при специфицировании интеграционных требований к моделям. Возможны и другие виды интерфейсов, «уточняющие» исходные интерфейсы из категории *SIG*. В то же время аспектная структура, выделяемая функтором *str*, не может служить интерфейсом для нетривиальных АО-моделей, а в некоторых случаях служит их дискретной структурой. Функтор *str* формально выражает «квинтэссенцию» аспектно-ориентированного расширения технологий проектирования систем, не сводимую к понятиям модульного подхода. Он сюръективен, т.е. любая аспектная структура реализуется в подходящей АО-модели.

Трансформации АО-моделей строятся в работе как двойственные к трассам: положим $tr\text{-}AO = (\mathrm{Ob}\, AO, mod^{-1}(\mathrm{Mor}\, tr\text{-}DESC))$, где *tr-DESC* – подкатегория в *c-DESC*, состоящая из всех *c-DESC*-объектов и всех трасс. В качестве конфигураций АО-моделей целесообразно выбрать *AO*-диаграммы, которые

[32] *Pratt V.R.* Modeling concurrency with partial orders // Intl. J. Parallel Programming. 1986. V. 15(1). P. 33–71.
[33] *Douence R. et al.* Trace-based aspects. In: Aspect-Oriented Software Development. Reading: Addison Wesley, 2004. P. 201–218.
[34] *Adámek J. et al.* Abstract and Concrete Categories. N. Y.: Wiley and Sons, 1990.

расширяют модульные конфигурации (т.е. находятся в классе $\mathbf{D}mod^{-1}(Conf)$) и копределы которых согласованно вычисляются на уровне модульных основ и аспектных структур (т.е. детерминируются функтором $\langle mod, str\rangle : AO \to c\text{-}DESC \times SIG$). Дополнительно налагается условие кооднородности (т.е. замкнутости относительно накачек *tr-AO*-морфизмами). Таким путем функтор *ai* с областью *AO порождает AO-технологию* над *AR* (она обозначается через $AO_{ai}(AR)$), если он удовлетворяет условиям (ii)-(v) определения 2.3 и, кроме того, существует функтор *si* такой, что $si \circ ai = int$.

Теорема 4.1. Любой фундаментальный функтор порождает АО-технологию над *AR*. □

Дискретная реализация модульных интерфейсов задается функтором дискретной разметки $mod^* : c\text{-}DESC \to AO : A \mapsto \langle A, 1_{sig(A)}\rangle, f \mapsto \langle f, sig(f)\rangle$.

Существуют технологии проектирования, аспектно-ориентированное расширение которых не привносит ничего существенно нового: в них функтор *mod* задает эквивалентность категорий *AO* и *c-DESC*. Такие технологии названы в работе *аспектно тривиальными*. Формальная технология аспектно тривиальна тогда и только тогда, когда любая разметка является изоморфизмом. Примером служат описанные выше технологии проектирования вычислительных систем CSD, CSD_*.

Аспектом (aspect) в АОП называется элементарный строительный блок аспектно-ориентированной программы, реализующий отдельный класс задач. Аспектная структура аспекта не может быть разрушена при интеграции в любую систему. Наиболее полное сохранение структуры обеспечивают действия, которые на уровне аспектных структур обладают обратимостью слева, т.е. возможностью идентифицировать аспектную структуру компонента в составе системы путем трассирования. Поэтому *AO*-морфизм, *str*-образ которого обратим слева, назван в работе *аспектным*, и дано следующее формальное определение аспекта:

Определение 4.8. АО-модель *A* называется *аспектом*, если любой *AO*-морфизм с областью *A* является аспектным. □

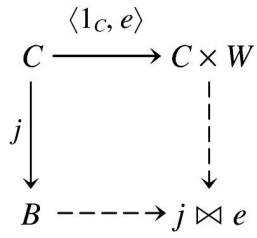
Чтобы проиллюстрировать элементарность аспектов, рассмотрен случай, когда категория *c-DESC* обладает терминальным объектом. В этом случае АО-модель является аспектом тогда и только тогда, когда ее аспектная структура изоморфна *sig*-образу терминального *c-DESC*-объекта. Например, в АО-технологии над *SM* аспект – это непустой сценарий, все события которого имеют одну и ту же метку, т.е. относятся к одному классу задач.

Процедура сборки системы из аспектов называется связыванием (weaving). В АОП она состоит в подключении программы, называемой советом (advice), к базовой программе (base) в заданных местах, называемых точками соединения (join point)[35]. Каждый раз, когда при исполнении базовой про-

[35] Aspect-Oriented Software Development. Reading: Addison Wesley, 2004.

граммы встречается точка соединения, вызывается совет. Поэтому он обычно выглядит как блок программного кода, охраняемый (guarded) условием, идентифицирующим точку соединения; начало блока служит точкой вызова совета (entry point). При связывании сначала (виртуально) создается достаточное количество копий совета, по одной на каждую точку соединения, с маркировкой соответствующих им точек вызова. Далее эти точки «склеиваются» друг с другом так, чтобы не разрушить аспектную структуру базы и совета. Для формальной записи правил связывания привлекается дополнительная АО-модель, называемая связкой (connector)[36], которая интегрируется с базой в точках соединения, а с советом – в точках вызова. Это позволяет выполнить связывание путем соединения, т.е. построения кодекартова квадрата специального вида:

Определение 4.9. *Аспектным связыванием* пары *AO*-морфизмов $j : B \leftarrow C \rightarrow W : e$, где B называется базой, W – советом, C – связкой, называется кодекартов квадрат пары $j : B \leftarrow C \rightarrow C \times W : \langle 1_C, e \rangle$ (*схемы связывания*), если он существует (в частности, существует произведение $C \times W$) и функтор *str* детерминирует как произведение $C \times W$, так и этот кодекартов квадрат. *Результатом связывания* называется вершина кодекартова квадрата, обозначается через $j \bowtie e$. □



В работе доказан ряд свойств связывания (для случаев, когда оно существует): единственность результата с точностью до изоморфизма, наличие неразрушающего аспектного вложения базы в результат, независимость результата привязывания нескольких взаимно независимых советов от порядка связывания, сохранение аспектной структуры базы при связывании с достаточно мелкими единицами аспектной декомпозиции. На подходящей подкатегории в **D***AO*, состоящей из всех диаграмм, изображающих спецификации связывания, построен функтор, сопоставляющий каждой спецификации результат ее связывания.

Реализация связывания требует специальных инструментов, таких как генераторы программного кода или мониторы исполнения программ. Но они могут не понадобиться для подключения аспектов, которые можно оформить единицами модульной архитектуры: объектами, таблицами в базе данных и т.д. Модуляризируемые АО-модели выделяются тем, что при интеграции с модулями они ведут себя так же, как модульные единицы. Возможности интеграции модулей в АО-модель определяются ее модульным интерфейсом, а АО-модели в модули – ее аспектной структурой. Поэтому модуляризация аспектов формализована в работе следующим образом: модуляризируемые АО-модели образуют полную подкатегорию в *AO*, а модули (*c-DESC-*

[36] *Pinto M. et al.* DAOP-ADL: an architecture description language for dynamic component and aspect-based development // Lecture Notes in Computer Sci. 2003. V. 2830. P. 118–137.

объекты) преобразуются в них посредством функтора, обладающего подходящими сопряженными (подобно функтору выделения интерфейсов).

Определение 4.10. Функтор $am : c\text{-}DESC \to m\text{-}AO$, где $m\text{-}AO$ – некоторая полная подкатегория в AO, называется *аспектно-ориентированным расширением* (АО-расширением) формальной технологии AR, если он обладает следующими сопряженными функторами:

- правый сопряженный am_* с тождественной единицей, причем $am_*(f) = mod(f)$ для любого $f \in \mathrm{Mor}\ m\text{-}AO$;
- левый сопряженный am^* с тождественной коединицей, причем $sig(am^*(f)) = str(f)$ для любого $f \in \mathrm{Mor}\ m\text{-}AO$.

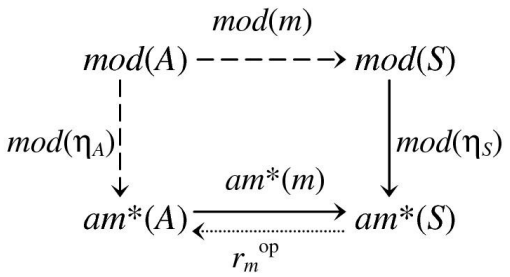
АО-расширение am называется:

- *тривиальным*, если оно является изоморфизмом;
- *наибольшим*, если любое АО-расширение $am' : c\text{-}DESC \to m\text{-}AO'$ удовлетворяет условию $\mathrm{Ob}\ m\text{-}AO' \subseteq \mathrm{Ob}\ m\text{-}AO$;
- *полным*, если категория $m\text{-}AO$ эквивалентна AO. □

АО-расширение совместимо и со сборкой систем (оно сохраняет копределы всех диаграмм), и с выделением интерфейсов (имеем $int(am(-)) = sig(-)$), и с трассированием (оно переводит трассу любой трассируемой трансформации в $tr\text{-}AO$-морфизм). Тривиальное АО-расширение существует у любой формальной технологии проектирования – это изоморфизм между $c\text{-}DESC$ и полной подкатегорией в AO с классом объектов $\{\langle A, 1_{sig(A)}\rangle \mid A \in \mathrm{Ob}\ c\text{-}DESC\}$, действующий как функтор mod^*. По существу (с точностью до естественного изоморфизма) любое АО-расширение совпадает с ним: модули ($c\text{-}DESC$-объекты) всегда переходят в АО-модели, история получения интерфейсов которых из классов задач путем трансформации утрачена (тривиальна). Таким образом, АО-расширение по существу однозначно определяется своей кообластью $m\text{-}AO$. Существуют структурируемые подтехнологии в $AO_{mod}(AR)$, обладающие категорией моделей $m\text{-}AO$ и функтором выделения интерфейсов am_*. Среди них существует наибольшая, которая названа в работе *модуляризируемой АО-технологией* над AR, порожденной АО-расширением am.

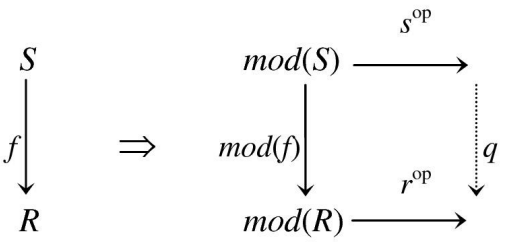
Пусть η – единица сопряжения $am^* \dashv am$, т.е. семейство $m\text{-}AO$-морфизмов вида $\eta_S : S \to am(am^*(S))$, $S \in \mathrm{Ob}\ m\text{-}AO$, определямых соотношением $am^*(\eta_S) = 1_{am^*(S)}$, так что для любого $m\text{-}AO$-морфизма $m : A \to S$ выполняется тождество $am(am^*(m)) \circ \eta_A = \eta_S \circ m$. Из него получается равенство $am^*(m) \circ mod(\eta_A) = mod(\eta_S) \circ mod(m)$, которое названо в работе *тождеством* (*am*-)*модуляризации* ввиду следующих соображений. Поскольку $c\text{-}DESC$-объект $am^*(S)$ представляет на модульном уровне аспектную структуру АО-модели S (в частности, $sig(am^*(S)) = str(S)$), морфизм $mod(\eta_S) : mod(S) \to am^*(S)$ служит (*am*-)*модуляризацией* ее аспектной структуры – канонической интеграцией модульной основы в модуляризированную аспектную структуру. Аспекты извлекаются из S путем трассирования их меток (обозначений

классов задач) вдоль модуляризации аспектной структуры. В работе показано, что такому трассированию отвечает ситуация, когда для морфизма $m : A \to S$, где A – извлекаемый аспект, морфизм $am^*(m)$ является правым обратным к трассе некоторой трансформации $r_m : am^*(A) \to am^*(S)$ (ср. пояснения к семантике трассирования), а тождество модуляризации представляет собой декартов квадрат – категорный аналог конструкции полного прообраза подмножества относительно отображения[37]. Таким образом, подаспекты определены как вложения (аспектные регулярные мономорфизмы) аспектов, модуляризация которых имеет универсальный характер.



Разбиение АО-модели на подаспекты описывает разделение ответственности (separation of concerns) – одну из ключевых процедур инженерии систем. Доказано, что аспекты являются атомарными единицами разделения ответственности – они не имеют собственных подаспектов. Каноническим способом модуляризации АО-модели и разделения ответственности служит восстановление трассируемой трансформации, породившей ее разметку:

Определение 4.14. *Экспликацией* (аспектной структуры) АО-модели $\langle A, l\rangle$ называется трассируемая трансформация s некоторого *r-DESC*-объекта в A такая, что $sig(s^{op}) = l$. *Экспликацией действия АО*-морфизма $f : S \to R$ (вдоль экспликаций s и r АО-моделей S и R, соответственно) называется *c-DESC*-морфизм q такой, что $q \circ s^{op} = r^{op} \circ mod(f)$. Экспликация s АО-модели S называется *универсальной*, если любой *АО*-морфизм с областью S эксплицируем вдоль s и любой экспликации своей кообласти. *Аспектным ядром* формальной технологии *AR* называется полная подкатегория *c-AO* в *AO*, состоящая из всех объектов, обладающих универсальной экспликацией. Формальная технология *AR* называется *аспектно универсальной*, если ее ядро совпадает с *AO*. *Экспликацией трансформации* АО-моделей g называется трансформация некоторых *r-DESC*-объектов, обладающая трассой, эксплицирующей действие *АО*-морфизма g^{op}. □



Теорема 4.2. Существует АО-расширение $ac : c\text{-}DESC \hookrightarrow c\text{-}AO : A \mapsto \langle A, 1_{sig(A)}\rangle, g \mapsto \langle g, sig(g)\rangle$, причем $ac^*(f)$ эксплицирует любой *c-AO*-морфизм f, и *ac*-модуляризация любого *c-AO*-объекта S представляет собой трассу его универсальной экспликации. □

[37] *Голдблатт Р.* Топосы. Категорный анализ логики. М.: Мир, 1983.

Следствие 4.2.1. Универсальная экспликация АО-модели, если она существует, определяется однозначно с точностью до изоморфизма. □

Следствие 4.2.2. АО-технология над аспектно универсальной формальной технологией, порожденная функтором *mod*, структурируема. □

Определение 4.15. *Ядерной АО-технологией* над произвольной формальной технологией проектирования *AR* называется модуляризируемая АО-технология над *AR*, порожденная функтором *ac*. □

Для формальных технологий, не обладающих свойством аспектной универсальности, в работе предложен более слабый подход к модуляризации аспектной структуры. В его основе лежит понятие частичного морфизма – это пара стрелок с общим началом, одна из которых является мономорфизмом и задает область определения частичного морфизма, а другая задает действие частичного морфизма на области определения[38].

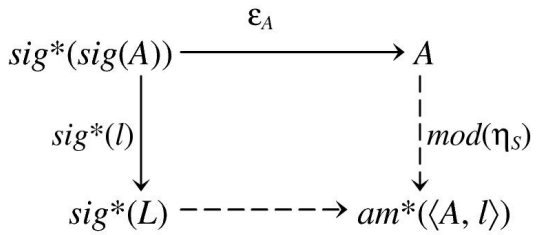
Определение 4.16. *Частичной модуляризацией* произвольной АО-модели $\langle A, l : sig(A) \to L\rangle$ называется следующий частичный *c-DESC*-морфизм из ее модульной основы A в дискретную реализацию аспектной структуры L:

$$\varepsilon_A : A \leftarrow sig^*(sig(A)) \to sig^*(L) : sig^*(l). \square$$

Здесь ε – коединица сопряжения $sig^* \dashv sig$, она состоит из мономорфизмов. Частичную модуляризацию можно рассматривать как диаграмму, состоящую из двух стрелок с общим началом, т.е. как конфигурацию соединения модульной основы АО-модели с дискретной реализацией ее аспектной структуры. Поэтому копредел такой диаграммы (если он существует) способен служить «аппроксимацией» модуляризации аспектной структуры:

Теорема 4.3. АО-технология $AO_{mod}(AR)$ структурируема тогда и только тогда, когда диаграмма частичной модуляризации любой АО-модели имеет копредел. □

Следствие 4.3.1. Если некоторая АО-модель содержится в кообласти некоторого АО-расширения, то диаграмма ее частичной модуляризации имеет копредел, сохраняемый функтором *sig*. Его можно выбрать так, чтобы ребро, параллельное действию частичной модуляризации, было модуляризацией аспектной структуры модели. □



Следствие 4.3.2. Модуляризация аспектной структуры АО-модели, если она существует, является регулярным эпиморфизмом. □

Критерию теоремы 4.3 удовлетворяет ряд частных случаев: когда технология *AR* аспектно тривиальна (так что действие любой частичной модуляризации является изоморфизмом), когда функтор *sig* является эквивалентно-

[38] *Adámek J. et al.* Abstract and Concrete Categories. N. Y.: Wiley and Sons, 1990.

стью категорий (так что коединица ε состоит из изоморфизмов), когда в категории *c-DESC* существуют кодекартовы квадраты (это имеет место в технологии моделирования сценариев *SM*, поскольку категория **Pos** кополна).

Семантика расширения технологий разработки систем аспектно-ориентированными приемами позволяет распознавать эти приемы в условиях, когда терминология АОП не используется. Примером служит синтез специализированных технологий специфицирования систем – краеугольный камень подхода MDE. В работе доказано, что функтор выделения интерфейсов, дополняющий формальную технологию конфигурирования до технологии специфицирования, представляет собой в точности ее эксплицируемую разметку в формальной технологии синтеза технологий конфигурирования ***SCONF***. Таким образом, *технологией проектирования технологий специфицирования* правомерно называть ядерную АО-технологию над ***SCONF***.

Обозначим через *AOSC* произвольную АО-технологию над ***SCONF***.

Теорема 4.4.

(i) Формальная технология конфигурирования *CS* может служить модульной основой аспекта в *AOSC* тогда и только тогда, когда категория ***desc***(*CS*) представляет собой предпорядок, имеющий наименьший элемент.

(ii) АО-модель над ***SCONF*** обладает экспликацией тогда и только тогда, когда она является формальной технологией специфицирования; любая ее экспликация универсальна.

(iii) Трансформация в *AOSC* обладает экспликацией тогда и только тогда, когда ее область и кообласть являются формальными технологиями специфицирования. □

Следствие 4.4.1. Ядерная АО-технология над ***SCONF*** имеет категорию моделей **SPEC** и функтор выделения интерфейсов ***conf***. □

В развитие этого результата построена формальная технология синтеза технологий проектирования. Для них технологии специфицирования не могут служить интерфейсами, поскольку функтор ***spec*** : **ARCH** → **SPEC**, «забывающий» категорию трансформаций, не унивалентен. Целесообразно извлекать интерфейсы из технологий проектирования таким же способом, каким функтор ***conf*** извлекает интерфейсы из технологий специфицирования – путем «забывания» функтора выделения интерфейсов (sic!). Поэтому интерфейсом технологии проектирования является любая тройка ⟨*c-DESC*, *Conf*, *r-DESC*⟩, удовлетворяющая условиям (i), (vi)-(viii) определения 2.3. Такие тройки называются *формализмами проектирования* (design formalisms)[39]. Конфигурации технологий проектирования отбираются из потенциала комплексируемости функтора, выделяющего формализмы из тех-

[39] *Lopes A., Fiadeiro J.L.* Revisiting the categorical approach to systems // Lecture Notes in Computer Sci. 2002. V. 2422. P. 426–440.

нологий проектирования, по критерию совместимости с конфигурациями технологий специфицирования. Трансформации технологий проектирования расширяют трансформации технологий специфицирования так, чтобы обеспечить трассируемость.

Например, функтор $mod : AO \to c\text{-}DESC$ порождает трансформацию технологии проектирования *AR* в АО-технологию над *AR* при условии, что технология *AR* аспектно полна (в том смысле, что сборка систем в ней не разрушает аспектную структуру) и все трансформации в ней трассируемы. Этому условию удовлетворяет, в частности, технология моделирования сценариев *SM*. Таким образом, конструкция АО-технологии совместима с аспектной структурой над ***SCONF***.

Глава 5 посвящена технологиям формального моделирования больших систем. В ней рассмотрена ситуация, характерная для подхода MDE, когда выбор технологии моделирования допускает определенный произвол, и требуется произвести его так, чтобы получить наибольший эффект (снижение затрат) от ее применения в жизненном цикле больших систем. Проще всего варьировать структуру трансформаций, поскольку технология специфицирования достаточно жестко определяется выбранным формальным методом моделирования: категория интерфейсов «естественным» образом выбирается среди полных корефлективных подкатегорий в категории моделей (строго говоря, среди их изоморфных образов, согласно условию (iii) определения 2.3), а конфигурации отбираются из потенциала комплексируемости корефлектора. Сложные технологии специфицирования синтезируются при помощи формализованных модульных и аспектно-ориентированных приемов в рамках технологии проектирования технологий специфицирования.

В работе рассмотрен выбор трансформаций по критерию способности трассировать включения компонентов в системы – действия по интеграции, не изменяющие внутреннюю структуру компонентов (и благодаря этому трассируемые «бесплатно»). Такие трансформации конструируются начиная с уровня интерфейсов: прежде всего выбирается класс L всех разметок. Трассы всех трансформаций находятся в классе $sig^{-1}(\mathrm{L})$, поэтому действия по интеграции, составляющие его, считаются наиболее сильно изменяющими структуру компонентов. Следовательно, включения, не разрушающие ее, представляют собой действия, «ортогональные» им.

На языке теории категорий условие ортогональности формулируется так: любое действие по интеграции однозначно (с точностью до изоморфизма) разлагается в композицию включения с $sig^{-1}(\mathrm{L})$-морфизмом. Пара классов морфизмов, порождающая разложения такого рода, называется факторизационной системой[40]. Классическим примером служит факторизационная система (Epi, Mono) в категории множеств **Set**, где мономорфизмы задают

[40] *Adámek J. et al.* Abstract and Concrete Categories. N. Y.: Wiley and Sons, 1990.

включения подмножеств, а любой эпиморфизм размечает свою область элементами кообласти. Ортогональность включений трассам означает, что на уровне интерфейсов любое действие по интеграции компонента в систему сводится к включению в нее результата некоторой разметки компонента. Таким образом, воздействие системного окружения на интерфейс компонента заключается в его «огрублении» – отмене некоторой трансформации, определяемой однозначно с точностью до изоморфизма. Дополнительно в целях обеспечения неразрушаемости накладываются условия дискретизируемости и кооднородности. Трансформации отбираются из класса $sig^{-1}(\mathrm{L})^{\mathrm{op}}$ по критерию способности трассировать включения:

Определение 5.1. Пусть $ff : C \to D$ – произвольный унивалентный функтор, M – произвольный класс C-морфизмов. C-морфизм $f : T \to S$ называется M-*инициальным*, если для любых M-морфизма $m : X \to S$ и D-морфизма $k : ff(X) \to ff(T)$ таких, что $ff(f) \circ k = ff(m)$, существует C-морфизм $k^+ : X \to T$ такой, что $ff(k^+) = k$. □

Определение 5.2. Формальная технология $SC = \langle c\text{-}DESC, Conf, sig\rangle$ *поддерживает* L-*разметки*, где L – некоторый класс SIG-морфизмов, если выполняются следующие условия:

(i) любой L-морфизм является ретракцией;

(ii) существует класс $\mathrm{M} \subseteq \mathrm{Mor}\ c\text{-}DESC$ такой, что пара $(sig^{-1}(\mathrm{L}), \mathrm{M})$ является факторизационной системой в c-$DESC$, и любой M-морфизм с дискретной кообластью имеет дискретную область;

(iii) класс $Conf$ замкнут относительно накачек $sig^{-1}(\mathrm{L})$-морфизмами.

При этом четверка $SC_{\mathrm{L}} = \langle c\text{-}DESC, Conf, sig, (\mathrm{Tr\ L})^{\mathrm{op}}\rangle$, где $\mathrm{Tr\ L} = (\mathrm{Ob}\ c\text{-}DESC, sig^{-1}(\mathrm{L}) \bigcap \mathrm{M\text{-}Init})$, называется L-*трансформационной технологией проектирования* над SC, M-морфизмы называются SC_{L}-*включениями*. □

В работе доказано, что SC_{L} является кооднородной формальной технологией проектирования с классом всех разметок L, в которой все трансформации трассируемы и даже обратимы справа, причем $\mathrm{RegMono}\ c\text{-}DESC \subseteq \mathrm{M} \subseteq \mathrm{Mor\text{-}Init} \subseteq \mathrm{M\text{-}Init}$, где M-Init – класс всех M-инициальных морфизмов. Трансформационные технологии хорошо поддаются трассированию вдоль аспектных трансформаций технологий специфицирования (в частности, они «опускаются» на уровень интерфейсов), а также аспектной ориентации:

Теорема 5.1. Пусть $SC' = \langle c\text{-}DESC', Conf', sig' : c\text{-}DESC' \to SIG'\rangle$ – произвольная формальная технология специфицирования, $\langle cm, sm\rangle : SC \to SC'$ – аспектная ***conf***-трасса. Если SC поддерживает L-разметки, то SC' поддерживает $sm(\mathrm{L})$-разметки, причем класс всех $SC'_{sm(\mathrm{L})}$-включений имеет вид $cm(\mathrm{M}) \circ \mathrm{Iso}\ c\text{-}DESC'$, где M – класс всех SC_{L}-включений. □

Следствие 5.1.1. Если SC поддерживает L-разметки, то тройка $SSIG = \langle SIG, sig \circ Conf, 1_{SIG}\rangle$ является формальной технологией специфицирования, поддерживающей L-разметки, для которой класс всех L-трансформаций совпадает с L^{op}, а класс всех включений – с $sig(\mathrm{M})$, где M – класс всех SC_{L}-

включений. Функтор *sig* индуцирует трансформацию L-трансформационной технологии $SSIG_{\mathrm{L}}$ в SC_{L}. □

Следствие 5.1.2. Если *SC* поддерживает L-разметки, то все *SIG*-объекты и все L-морфизмы образуют подкатегорию в *SIG*, содержащую все *SIG*-изоморфизмы. □

Следствие 5.1.3. Пусть $SC' = \langle c\text{-}DESC', Conf', sig' : c\text{-}DESC' \to SIG' \rangle$ – произвольная формальная технология специфицирования, $\langle cm, sm \rangle : SC \to SC'$ – **SPEC**-морфизм такой, что *cm* – ***desc***-разметка, *sm* – изоморфизм категорий. Если *SC* поддерживает L-разметки и *SC*' поддерживает L'-разметки, то соотношение $sm(\mathrm{L}) \subseteq \mathrm{L}'$ выполняется тогда и только тогда, когда тройка $\langle cm, sm, cm(-^{\mathrm{op}})^{\mathrm{op}} \rangle$ представляет собой **ARCH**-морфизм из SC_{L} в $SC'_{\mathrm{L}'}$. □

Следствие 5.1.4. Технология специфицирования $\boldsymbol{spec}(AO_{int}(SC_{\mathrm{L}}))$ поддерживает L-разметки, причем L-трансформационная технология $\boldsymbol{spec}(AO_{int}(SC_{\mathrm{L}}))_{\mathrm{L}}$ является подтехнологией в $AO_{int}(SC_{\mathrm{L}})$. Морфизм AO-моделей над SC_{L} является включением тогда и только тогда, когда функторы *mod* и *str* переводят его во включения (т.е. в M-морфизм и в *sig*(M)-морфизм, соответственно). □

В работе показано, что трансформационной является типовая технология моделирования данных. Наиболее общая формальная модель массива данных – это множество хранящихся в нем информационных элементов. Действие по интеграции массивов – это в точности любое отображение множеств. Любая диаграмма множеств и отображений является допустимой конфигурацией. Получается формальная технология конфигурирования $DC = \langle \mathbf{Set}, \mathrm{Ob}\ \mathbf{D}(\mathbf{Set}) \rangle$. Категория **Set** не содержит полных корефлективных подкатегорий с унивалентным корефлектором, не эквивалентных ей, так что переход к технологии специфицирования осуществляется формально, при помощи функтора ***conf****: получается технология $DS = \boldsymbol{conf}^*(DC) = \langle \mathbf{Set}, \mathrm{Ob}\ \mathbf{D}(\mathbf{Set}), 1_{\mathbf{Set}} \rangle$ (специфицирование моделей данных фактически сводится к конфигурированию). Частные информационные технологии оперируют с разнообразными специальными видами множеств, поэтому им отвечают подтехнологии в *DS*. Например, таблица в реляционной базе данных представляет собой подмножество прямого произведения основных множеств типов атрибутов (domains)[41]. Интеграция таблиц выполняется посредством внешних ключей – отображений множеств записей.

Технология *DS* поддерживает два класса разметок: класс Bij всех биекций множеств и класс Surj всех сюръекций. В Bij-трансформационной технологии над *DS* класс всех трансформаций совпадает с Bij, поэтому она фактически воспроизводит специфицирование интерфейсов (и не имеет нетривиального аспектно-ориентированного расширения). С классом Surj дело об-

[41] *Бениаминов Е.М.* Алгебраические методы в теории баз данных и представлении знаний. М.: Научный мир, 2003.

стоит иначе: в DS_{Surj} класс всех трансформаций состоит из всех тотальных антифункциональных бинарных отношений, а класс всех включений – из всех инъекций множеств. Обозначим через *DM* технологию DS_{Surj}. Категория АО-моделей над *DM* представляет собой полную подкатегорию **Surj** в $\mathbf{Set}^{\mathbf{2}}$, класс объектов которой состоит из всех сюръекций множеств. Поэтому технология *DM* аспектно универсальна: разметка любого **Surj**-объекта представляет собой трассу его единственной экспликации. Отсюда получается прямой способ реализации разметки произвольного массива данных: разметка задается действием по его интеграции в массив-справочник классов задач.

В работе построена каноническая АО-технология над *DM* – это $AO_{mod}(DM)$, совпадающая с ядерной АО-технологией над *DM* и обозначаемая через ***AODM***. Пусть $isurj : \mathbf{Surj} \hookrightarrow \mathbf{Set}^{\mathbf{2}}$ – вложение. Имеем

$$\boldsymbol{AODM} = \langle \mathbf{Surj}, \mathrm{Ob}\ \mathbf{D}(\mathbf{Surj}), dom \circ isurj : \mathbf{Surj} \to \mathbf{Set}, ((\mathrm{Ob}\ \mathbf{Set}, \mathrm{Surj})^{\mathbf{2}})^{\mathrm{op}} \rangle.$$

Любой функтор, порождающий АО-технологию над *DM*, по существу (с точностью до эквивалентности категорий) совпадает либо с $dom \circ isurj$, либо с $1_{\mathbf{Surj}}$. Технология ***AODM*** скоординирована и структурируема.

Связывание, как и разделение ответственности, в технологии моделирования данных существует всегда, причем всегда является частным случаем модульной компоновки. Связывание имеет прозрачный смысл для реляционных таблиц: здесь его спецификация описывает добавление атрибута типа «список ссылок на совет» (отношение «многие-ко-многим»), с использованием связки в качестве источника ссылочных ключей. Добавление атрибута в результате связывания не изменяет структуру базовой таблицы, поэтому не требует значительных затрат. Путем связывания целесообразно синтезировать информационные модели объектов, вступающих в многообразные и изменчивые отношения. В свою очередь, разделение ответственности в моделях данных сводится к распараллеливанию – ему отвечает наиболее «мелкая» сумма в технологии ***AODM***. Для реляционной таблицы оно может быть реализовано путем секционирования[42] (partitioning) по значению атрибута-ссылки на справочник классов задач. Это приводит к повышению производительности без дополнительных затрат труда программистов: исключается конкуренция компонентов, автоматизирующих разные классы задач, за доступ к общей таблице.

Технология моделирования сценариев *SM* также является трансформационной: технология специфицирования $SS = \boldsymbol{spec}(SM)$ поддерживает L-разметки тогда и только тогда, когда L = Surj, причем класс всех SS_{Surj}-включений совпадает с классом всех регулярных **Pos**-мономорфизмов, и $SS_{\mathrm{Surj}} = SM$. Трасса трансформации сценариев разбивает свою область на совокупность прообразов точек, хорошо упорядоченную (well-ordered) в том смысле, что

[42] *Туманов В.Е.* Основы проектирования реляционных баз данных. М.: Бином, 2010.

для любых x, y, z и u таких, что $t(x) = t(y) \neq t(z) = t(u)$, условие $x \leq z$ влечет $y \leq u$. Такая трактовка трансформаций предлагалась в литературе[43], однако их взаимосвязь с регулярными **Pos**-мономорфизмами, состоящая в том, что все трассы образуют класс (Epi **Pos**) ⋂ (RegMono **Pos**)-Init, не была выявлена.

Пусть *AOSM* – категория АО-моделей, порожденная технологией *SM*. Она (ко)полна, и более того, эквивалентна топологической категории над **Pos**. Тем не менее связывание пары *AOSM*-морфизмов $j : B \leftarrow C \rightarrow W : e$ существует не всегда, а тогда и только тогда, когда для любых $x, y \in C$ условия $j(x) = j(y)$ и $x \leq y$ влекут $l(v) = l(e(x))$, где l – разметка сценария W, v – его произвольный элемент, удовлетворяющий условию $e(x) \leq v \leq e(y)$. Разделение ответственности сценария, в свою очередь, существует тогда и только тогда, когда его аспекты хорошо упорядочены, т.е. когда он обладает экспликацией (которая всегда универсальна). В работе выявлены свойства классов разделений ответственности, естественным образом возникающих при распараллеливании: разделение на взаимно независимые подаспекты и на линейно упорядоченные подаспекты.

Рассмотрен пример связывания сценариев при проектировании информационно-управляющих систем с применением событийной модели. Здесь сценарий основного процесса обработки событий B выглядит так:

$$\textit{регистрация} \rightarrow \textit{сохранение} \rightarrow \textit{анализ} \rightarrow \textit{воздействие}.$$

Инфраструктурные аспекты, такие как ведение информационной модели объекта (паспорта), комплексируются с ним путем связывания в сложные сценарии, не допускающие разделение ответственности. Так, необходимость изменить паспорт объекта может возникнуть при сохранении события в базу данных системы, например если оно сигнализирует о замене прибора. Другие фрагменты паспорта могут измениться по результатам анализа, например если зарегистрировано потребление энергии единицей оборудования, не связанной в паспорте с питающим центром. Поведение аспекта изменения паспорта W упрощенно описывается помеченным сценарием из двух последовательных событий, относящихся к одному классу задач:

$$\textit{изменение}_{\textit{запрос}} \rightarrow \textit{изменение}_{\textit{внес}}.$$

Его связка с базовым сценарием имеет вид дискретного сценария $\mathbf{1} \coprod \mathbf{1}$: морфизм j представляет собой его биекцию на подмножество $\{\textit{сохранение}, \textit{анализ}\} \subseteq B$, а e – постоянное отображение на элемент $\textit{изменение}_{\textit{запрос}} \in W$. Связывание существует и порождает следующий помеченный сценарий:

$$\begin{array}{ccccc} \textit{регистрация} \rightarrow \textit{сохранение} & & \rightarrow \textit{анализ} & \rightarrow \textit{воздействие} \\ \downarrow & & \downarrow & \\ \textit{сохранение}_{\textit{внес}} & & \textit{анализ}_{\textit{внес}}, & \end{array}$$

[43] *Glabbeek R.J. van, Goltz U.* Refinement of actions and equivalence notions for concurrent systems // Acta Informatica. 2000. V. 37, N 4–5. P. 229–327.

где индексом *внес* снабжены события соответствующих классов задач, фиксирующие внесение изменений в паспорт объекта. Здесь наглядно видно, как процедуры паспортизации ассимилируются основным процессом: метка аспекта изменения паспорта в этом сценарии отсутствует – она «сливается» с метками точек соединения. В то же время каждый акт изменения паспорта приобретает пометку инициировавшей его задачей, которую можно сохранить вместе с изменяемыми данными (при условии, что хранилище данных паспорта спроектировано на базе технологии ***AODM***). В реальных системах такая процедура связывания позволяет рассеивать задачу ведения паспорта по сотням процессов-источников данных, с обеспечением трассирования каждого элемента к задачам, вызвавшим его изменение. В результате достигается практически полная актуальность и достоверность паспорта, причем со значительным снижением затрат труда по сравнению с традиционными «ручными» регламентными процедурами ведения паспорта.

Так устанавливается взаимосвязь между технологиями аспектно-ориентированного моделирования данных и процессов, обуславливающая построение совместных моделей. Указанная взаимосвязь формализована в работе в виде **ARCH**-морфизма из АО-технологии над *SM* в ***AODM***, который задает перенос разметки сценариев на массивы данных, естественный относительно и сборки систем, и выделения интерфейсов, и трансформаций. Этот **ARCH**-морфизм индуцируется функтором $asp : AOSM \to \mathbf{Surj}$.

Таким образом, при помощи теоретико-категорных конструкций, предложенных в работе, обеспечивается рациональность типовых проектных решений. Ряд решений был апробирован в сфере автоматизации больших объектов топливно-энергетического комплекса (ТЭК) при создании программной платформы «Энергиус». Был проведен цикл инженерии предметной области ТЭК, в ходе которого был построен большой комплекс моделей, включающий онтологию предметной области, модели процессов и организационных структур, формы документов и видеокадров, протоколы информационного обмена и расчетные алгоритмы. Онтология была основана на стандартах IEC 61970 (Common Information Model – CIM), ГОСТ Р 8.596-2002 «Метрологическое обеспечение измерительных систем. Основные положения» и других нормативных источниках. Был выделен нормированный перечень классов задач ТЭК, собранный в справочник – источник меток элементов данных и шагов процессов. Сущности предметной области были связаны в единую аспектно-ориентированную модель данных посредством отношений вида «многие-ко-многим», помеченных классами задач. Из этой модели была автоматически сгенерирована база данных платформы «Энергиус».

Справочник классов задач послужил источником меток для событий, порождаемых компонентами в ходе функционирования системы. Реализован компонент ведения единого журнала событий, с которым связаны (в смысле АОП) многочисленные обработчики событий: средства изменения паспорта

объекта, расчета разнообразных показателей, проверки соответствия фактического хода автоматизируемых процессов плановому, оповещения участников процессов и т.д. Данные, формируемые обработчиками, снабжаются ссылками на породившие их события. Так было обеспечено сквозное трассирование, позволившее снизить затраты на внесение изменений, проверку их корректности и согласованости.

Компоненты расчета показателей описываются проблемно-ориентированными моделями вычислений, в которых совокупность примитивов и вид вычисляемых функций задаются паспортом объекта. Традиционные «ручные» процедуры составления, верификации и актуализации программного кода этих моделей требуют значительных затрат труда, растущих пропорционально масштабу объекта. В целях снижения затрат, в соответствии с подходом MDE программная реализация расчетных компонентов автоматически генерируется по наполнению паспорта в виде динамически подгружаемых модулей. Генератор поставляется в составе платформы «Энергиус», поскольку он вызывается при изменениях паспорта для обновления реализаций изменившихся алгоритмов. Генератор обходит граф участка энергетической сети, показатели которого требуется рассчитать, и порождает корректный оптимизированный программный код для вычисления их значений. Порождаемые модули развертываются в среде распределенных вычислений, что позволяет обеспечить информационную поддержку принятия решений в темпе процесса оперативно-диспетчерского управления. Процесс проектирования генератора расчетных компонентов с применением теоретико-категорных методов, предложенных в работе, выглядит следующим образом.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

1. Для задачи управления жизненным циклом больших информационно-управляющих систем с целью уменьшения затрат при соблюдении ограничений на допустимые значения показателей качества, показана возможность решения путем привлечения масштабируемых технологий инженерии предметной области, разработки, управляемой моделями, распределенных вычислений, аспектно-ориентированного подхода.

2. Построен и теоретически обоснован аппарат для формального анализа и синтеза технологий проектирования систем на основе теории категорий и концепции формальной технологии, предоставляющий методы подбора рациональных архитектурных решений, интеграционных интерфейсов, способов трансформации результатов проектирования.

3. Построена и теоретически обоснована формальная технология проектирования вычислительных компонентов информационно-управляющих систем путем отображения алгоритмов на архитектуру вычислительной среды, включая редукцию на конечные множества доступных ресурсов, подбор рациональных средств управления потоками вычислений, рациональное комплексирование вычислительных компонентов в распределенные системы, оценку сложности вычислений с привлечением аппарата конечнозначной логики Лукасевича.

4. Построена и теоретически обоснована универсальная теоретико-категорная семантика расширения модульных технологий проектирования систем аспектно-ориентированными приемами и инструментами, с обеспечением трассирования результатов проектирования к классам задач и разделения ответственности путем экспликации аспектной структуры на уровень модулей.

5. Доказано, что разметка данных и сценариев поведения систем классами задач представляет собой частный случай перехода от модульного проектирования к аспектно-ориентированному и что необходимым и достаточным условием существования аспектного связывания помеченных сценариев является совместимость с конкуррентностью, а разделения ответственности – хорошая упорядоченность аспектов.

6. Доказано, что оснащение компонентов интеграционными интерфейсами, предназначенными для сборки систем, представляет собой частный случай перехода от модульного проектирования к аспектно-ориентированному, что позволяет применять аспектно-ориентированный подход для повышения эффективности синтеза технологий создания систем.

7. Разработаны модели данных, процессов, вычислений и других составляющих предметной области топливно-энергетического комплекса, на базе которых при помощи предложенных в работе приемов рационально постро-

ены большие системы диспетчерского управления, интеллектуального учета электроэнергии, управления энергосбережением.

СПИСОК ПУБЛИКАЦИЙ ПО ТЕМЕ ДИССЕРТАЦИИ

Публикации в изданиях, рекомендованных ВАК для представления результатов докторских диссертаций:

1. *Ковалёв С.П.* Архитектура времени в распределенных информационных системах // Вычислительные технологии. 2002. Т. 7, №6. С. 38–53.
2. *Ковалёв С.П.* Аналитические модели машинной арифметики // Сибирский журнал индустриальной математики. 2003. Т. 6, №3(15). С. 88–102.
3. *Ковалёв С.П.* Логика Лукасевича как архитектурная модель арифметики // Сибирский журнал индустриальной математики. 2003. Т. 6, №4(16). С. 32–50.
4. *Ковалёв С.П.* Применение формальных методов для обеспечения качества вычислительных систем // Вестник Новосибирского государственного университета. Серия: Математика, механика, информатика. 2004. Т. 4, №2. С. 49–74.
5. *Ковалёв С.П.* Математические основания компьютерной арифметики // Математические труды. 2005. Т. 8, №1. С. 3–42.
6. *Загоруйко Н.Г., Гусев В.Д., Завертайлов А.В., Ковалёв С.П., Налётов А.М., Саломатина Н.В.* Система ONTOGRID для автоматизации процессов построения онтологий предметных областей // Автометрия. 2005. Т. 41, №5. С. 13–25.
7. *Ковалёв С.П.* Алгебраический подход к проектированию распределенных вычислительных систем // Сибирский журнал индустриальной математики. 2007. Т. X, №2(30). С. 70–84.
8. *Ковалёв С.П.* Применение онтологий при разработке распределенных автоматизированных информационно-измерительных систем // Автометрия. 2008. Т. 44, №2. С. 41–49.
9. *Кузнецов А.А., Ковалёв С.П.* Тестирование и мониторинг в распределенных автоматизированных системах технологического управления // Вычислительные технологии. 2009. Т. 14, №4. С. 57–69.
10. *Ковалёв С.П.* Формальный подход к аспектно-ориентированному моделированию сценариев // Сибирский журнал индустриальной математики. 2010. Т. 13, №3. С. 30–42.
11. *Андрюшкевич С.К., Ковалёв С.П.* Интеллектуальный мониторинг распределенных технологических объектов с использованием информационных моделей состояния // Известия Томского политехнического университета. 2010. Т. 317, №5. Управление, вычислительная техника и информатика. С. 35–39.

12. *Ковалёв С.П., Пароджанов С.С.* Концепция создания автоматизированной системы мониторинга и управления энергоэффективностью на объектах города Москвы // Информационно-измерительные и управляющие системы. 2011. Т. 9, №6. С. 50–58.

13. *Андрюшкевич С.К., Ковалёв С.П.* Динамическое связывание аспектов в крупномасштабных системах технологического управления // Вычислительные технологии. 2011. Т. 16, №6. С. 3–12.

14. *Трегубов А.М., Ковалёв С.П.* Особенности проектирования систем мониторинга и управления энергосбережением // Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2012. Т. 10, №3. С. 80–91.

15. *Ковалёв С.П.* Диаграммное описание комплексирования программных систем // Вестник Новосибирского государственного университета. Серия: Математика, механика, информатика. 2012. Т. 12, №3. С. 103–126.

16. *Ковалёв С.П.* Системный анализ жизненного цикла больших информационно-управляющих систем // Автоматика и телемеханика. 2013. №9. С. 98–118.

17. *Ковалёв С.П.* Семантика аспектно-ориентированного моделирования данных и процессов // Информатика и ее применения. 2013. Т. 7, Вып.3. С. 70–80.

Учебные пособия:

18. *Ковалёв С.П.* Формальный подход к разработке программных систем. Учебное пособие. Новосибирск: НГУ, 2004.

Патенты:

19. *Андрюшкевич С.К., Золотухин Е.П., Ковалёв С.П.* Многоуровневая автоматизированная система управления производственно-технологическими процессами с управлением затратами на основе мониторинга, анализа и прогноза состояния технологической инфраструктуры нефтегазодобывающего предприятия. Патент на изобретение №2435188, ноябрь 2011.

Свидетельства о регистрации программ для ЭВМ:

20. *Ковалёв С.П., Андрюшкевич С.К., Яковченко К.Н.* Система автоматизации оперативно-диспетчерского документооборота «Энергиус-Диспетчер». Свидетельство о государственной регистрации программы для ЭВМ №2009611387, март 2009.

21. *Ковалёв С.П., Андрюшкевич С.К., Гуськов А.Е.* Интеграционная платформа учета и управления энергообеспечением «Энергиус». Свидетельство о государственной регистрации программы для ЭВМ №2009613359, июнь 2009.

22. *Ковалёв С.П., Андрюшкевич С.К.* Автоматизированная система учета энергоносителей «Энергиус-Учёт». Свидетельство о государственной регистрации программы для ЭВМ №2010613308, май 2010.

23. *Андрюшкевич С.К., Золотухин Е.П., Ковалёв С.П.* Экспериментальный образец программного комплекса системы оперативного мониторинга технологической инфраструктуры «ЭО СОМТИ» (ЭО ПК СОМТИ). Свидетельство о государственной регистрации программы для ЭВМ №2010613850, июнь 2010.

24. *Ковалёв С.П., Андрюшкевич С.К.* Автоматизированная система ведения нормативно-справочной информации в области энергетики «Энергиус-НСИ». Свидетельство о государственной регистрации программы для ЭВМ №2011613800, май 2011.

Прочие публикации:

25. *Belov S.D., Bredikhin S.V., Kovalyov S.P., Kulagin S.A., Musher S.L., Nikultsev V.A., Scherbakova N.G., Shabalnikov I.V., Shokin Yu.I.* Siberia: Internet is Coming // Proc. 7th Joint European Networking Conference JENC7. Budapest, 1996. P. 173/1–173/4.

26. *Belov S.D., Bredikhin S.V., Kovalyov S.P., Kulagin S.A., Musher S.L., Nikultsev V.A., Scherbakova N.G., Shabalnikov I.V., Shokin Yu.I.* Siberia: Putting the Virgin Lands to the Internet Plough // Proc. 8th Joint European Networking Conference JENC8. Edinburg, 1997. P. 523/1–523/4.

27. *Belov S.D., Bredikhin S.V., Kovalyov S.P., Kulagin S.A., Musher S.L., Scherbakova N.G., Shabalnikov I.V.* The emerging Internet landscape in Siberia // Computer Networks. 1998. Vol. 30. P. 1657–1662.

28. *Ковалёв С.П.* Принципы профессионального программирования // 3 Международная конференция «Смирновские чтения». М.: ИФРАН, 2001. С. 42–44.

29. *Завертайлов А.В., Ковалёв С.П.* Система поддержки деятельности распределённых экспертных групп по разработке онтологий предметных областей // Труды Международной конференции по вычислительной математике «МКВМ-2004». Рабочие совещания. Новосибирск: ИВМиМГ СО РАН, 2004. С. 56–65.

30. *Ковалёв С.П., Яковченко К.Н.* Организация информационных порталов на основе канальной интеграции // Труды Международной конференции по вычислительной математике «МКВМ-2004». Рабочие совещания. Новосибирск: ИВМиМГ СО РАН, 2004. С. 66–72.

31. *Новоселов Д.Ю., Ковалёв С.П.* Аспектно-ориентированный подход к созданию GRID приложений // Труды Международной конференции по вычислительной математике «МКВМ-2004». Рабочие совещания. Новосибирск: ИВМиМГ СО РАН, 2004. С. 95–102.

32. *Ковалёв С.П.* Алгебраические спецификации в экстремальном программировании // Материалы Международной конференции «Алгебра, логика и кибернетика-2004». Иркутск: ИГПУ, 2004. С. 155–157.

33. *Ковалёв С.П., Андрюшкевич С.К.* Автоматизированный инструмент компонентной интеграции информационных порталов // Труды Всероссий-

ской научной конференции «Научный сервис в сети Интернет». М.: МГУ, 2004. С. 98–99.

34. *Kovalyov S.P., Ozhiganova E.A.* Formal Modeling of Software Quality with xNoFun Language // Proc. Second IASTED International Conference on Automation, Control and Information Technology ACIT-2005. Software Engineering. Novosibirsk, 2005. P. 48–53.

35. *Новоселов Д.Ю., Ковалёв С.П.* LDAP-каталог СО РАН // Вычислительные технологии. 2005. Т. 10. Специальный выпуск: Труды IX рабочего совещания по электронным публикациям El-Pub2004. С. 102–107.

36. *Загоруйко Н.Г., Гусев В.Д., Завертайлов А.В., Ковалёв С.П., Налётов А.М., Саломатина Н.В.* Система «ONTOGRID» для построения онтологий // Компьютерная лингвистика и интеллектуальные технологии: Труды международной конференции «Диалог'2005». М.: Наука, 2005. С. 146–152.

37. *Загоруйко Н.Г., Гусев В.Д., Завертайлов А.В., Ковалёв С.П., Налётов А.М., Саломатина Н.В.* Автоматизация процессов построения онтологий // Proc. XI International Conference “Knowledge-Dialogue-Solution” (KDS-2005). Vol. 1. Varna, 2005. P. 53–59.

38. *Гордов Е.П., Ковалёв С.П., Молородов Ю.И., Федотов А.М.* Web-система управления знаниями об окружающей среде // Вычислительные технологии. Т. 10, ч.2. Специальный выпуск: Труды Международной конференции и школы молодых ученых «Вычислительные и информационные технологии для наук об окружающей среде» (CITES-2005). Новосибирск, 2005. С. 12–19.

39. *Kovalyov S.P., Molorodov Yu.I.* Data integration methodology for atlas “Atmospheric aerosols of Siberia” // Proc. SPIE. 2005. Vol. 6160. Part I: International Conference “Molecular spectroscopy and atmospheric radiative processes”. P. 9–14.

40. *Kovalyov S.P.* Architecture of distributed information-computing system for exploring atmospheric aerosol // Proc. SPIE. 2005. Vol. 6160. Part I: International Conference “Molecular spectroscopy and atmospheric radiative processes”. P. 21–26.

41. *Ковалёв С.П.* Применение логики Лукасевича для разработки алгоритмов. В кн.: Логические исследования. Вып. 12. М.: Наука, 2005. С. 194–206.

42. *Гуськов А.Е., Дубров И.С., Ковалёв С.П., Молородов Ю.И., Прокопов Н.А., Сударикова И.А.* Принципы построения распределённой среды атласа «Атмосферные аэрозоли Сибири» // Вычислительные технологии. Т. 11. Специальный выпуск: Труды X Российской конференции с участием иностранных ученых «Распределенные информационно-вычислительные ресурсы» DICR-2005. Новосибирск, 2005 г. С. 77–87.

43. *Ковалёв С.П.* Аспектное проектирование интегрированных систем управления энергопотреблением // Сборник материалов II международной

научно-технической конференции «Новые информационные технологии в нефтегазовой отрасли и образовании». Тюмень: ТюмГНГУ, 2006. С. 97–99.

44. *Kovalyov S.P.* Model-theoretic methods of analysis of computer arithmetic // Proc. 9th Asian Logic Conference “Mathematical Logic on Asia”. Singapore: World Scientific Publishing Co., 2006. P. 145–155.

45. *Барахнин В.Б., Клименко О.А., Ковалёв С.П.* Сбор и систематизация информации для портала математических ресурсов MATHTREE // Труды международной конференции «Вычислительные и информационные технологии в науке, технике и образовании». Т. II. Павлодар: ТОО НПФ «ЭКО», 2006. С. 381–389.

46. *Ковалёв С.П.* Полупримальные модели компьютерных вычислений // Международная конференция «Алгебра и ее приложения». Тезисы докладов. Красноярск: ИВМ СО РАН, 2007. С. 73–74.

47. *Ковалёв С.П., Прокопов Н.А.* Автоматизация процессов измерения физико-химических величин на базе онтологии // Вычислительные технологии. 2007. Т. 12, Специальный выпуск 1. С. 79–86.

48. *Андрюшкевич С.К., Ковалёв С.П.* Опыт адаптации стандартных информационных моделей для распределенных объектов технологического управления // Высокие технологии, фундаментальные исследования, образование: сборник трудов Седьмой международной научно-практической конференции «Исследование, разработка и применение высоких технологий в промышленности». Т. 1. СПб.: Изд-во Политехнического университета, 2009. С. 56–57.

49. *Кузнецов А.А., Ковалёв С.П.* Мониторинг состояния крупномасштабных систем технологического управления // Высокие технологии, фундаментальные исследования, образование: сборник трудов Седьмой международной научно-практической конференции «Исследование, разработка и применение высоких технологий в промышленности». Т. 1. СПб.: Изд-во Политехнического университета, 2009. С. 101–103.

50. *Ковалёв С.П.* Аспектно-ориентированный подход к проектированию систем мониторинга крупномасштабных объектов // Проблемы информатики. 2009. №3(4). С. 5–18.

51. *Kovalyov S.P.* Modeling aspects by category theory // Proc. 9th Workshop on Foundations of Aspect-Oriented Languages. Rennes, France, 2010. P. 63–68.

52. *Andryushkevich S.K., Kovalyov S.P.* Distributed plants intelligent monitoring using information models of states // Proc. IASTED International Conference on Automation, Control and Information Technology ACIT-2010. Control, Diagnostics, and Automation. Novosibirsk, 2010. P. 250–255.

53. *Андрюшкевич С.К., Журавлев С.С., Золотухин Е.П., Ковалёв С.П., Окольнишников В.В., Рудометов С.В.* Разработка системы мониторинга с использованием имитационного моделирования // Проблемы информатики. 2010. №4. С. 65–75.

54. *Андрюшкевич С.К., Ковалёв С.П.* О формировании аспектно-ориентированного связывания производственных задач в системах технологического управления // Материалы XIII Российской конференции «Распределенные информационные и вычислительные ресурсы» DICR'2010. Новосибирск, 2010. С. 32.

55. *Золотухин Е.П., Ковалёв С.П. Андрюшкевич С.К., Васильева Е.С., Воробьева Д.Б., Долгих Е.В., Кислицына Т.А., Лапшинов М.Ф., Окольнишников В.В., Романов Р.А., Шульженко Л.М.* Разработка интеллектуальной системы пространственно-технологического мониторинга на базе глобального спутникового позиционирования с целью повышения энергоэффективности и экологической безопасности существующих методов добычи углеводородов. Отчет о НИР по государственному контракту № 02.514.11.4126 от 30.09.2010 г. Минобрнауки РФ. № гос. рег. 1200962854. Новосибирск: КТИ ВТ СО РАН, 2010.

56. *Трегубов А.М., Ковалёв С.П.* Системы мониторинга и управления энергосбережением – новый класс крупномасштабных систем // Сборник статей XI Международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности». Т. 1. СПб.: Изд-во Политехнического университета, 2011. С. 132.

57. *Пароджанов С.С., Ковалёв С.П.* Система комплексного управления проектами в области энергосбережения // Научная сессия НИЯУ МИФИ-2012. Аннотации докладов. Т. 2. Проблемы фундаментальной науки. Стратегические информационные технологии. М.: НИЯУ МИФИ, 2012. С. 354.

58. *Андрюшкевич С.К., Ковалёв С.П., Кубышкин А.С., Трегубов А.М.* Проблемы автоматизации управления процессами розничного рынка электроэнергии // Труды XIV Международной конференции «Проблемы управления и моделирования в сложных системах» ПУМСС-2012. Самара: СамНЦ РАН, 2012. С. 376–386.

59. *Ковалёв С.П.* Применение аспектно-ориентированного подхода для автоматизации крупномасштабных объектов и процессов управления // Труды VI Международной конференции «Управление развитием крупномасштабных систем» MLSD'2012. Т. II. М.: ИПУ РАН, 2012. С. 314–323.

60. *Ковалёв С.П.* Категория вычислительных систем // Международная конференция «Алгебра и логика: теория и приложения». Тезисы докладов. Красноярск: СФУ, 2013. С. 64–66.