УДК 519.68

С. П. Ковалёв

ДИАГРАММНОЕ ОПИСАНИЕ КОМПЛЕКСИРОВАНИЯ ПРОГРАММНЫХ СИСТЕМ

В работе представлен теоретико-категорный подход к формализации процессов разработки программных систем, позволяющий единообразно описать многие известные технологии программирования. В качестве отправной точки для выработки подхода использована конструкция формальной дисциплины проектирования, введенной Х. Фиадейро и его учениками. Выделен ряд классов формальных дисциплин, встречающихся в практике программирования. На языке теории категорий сформулированы и исследованы задачи синтеза систем: применение шаблонов комплексирования, выявление оптимальной архитектуры, выбор интеграционных интерфейсов, распараллеливание, покомпонентная трансформация систем, разработка специализированных технологий комплексирования систем. Обосновано, что любой акт комплексирования комбинируется из трех приемов: загрузка, подстановка и соединение.

Ключевые слова: интеграция систем, категория диаграмм, формальная дисциплина проектирования, выявление архитектуры, мереология.

Введение

Комплексированием называется сборка сложных систем из отдельных составных частей (компонентов). Акты комплексирования наглядно описываются диаграммами — схемами, состоящими из обозначений составных частей и действий по их соединению. В традиционной технике для этого применяются сборочные чертежи, смысл которых однозначно определяется физической природой систем. Однако для инженерии программных систем характерно многообразие подходов к выделению частей (декомпозиции), отражающих различные точки зрения на автоматизируемую предметную область. Они определяют различную семантику единиц комплексирования и правил взаимодействия между ними в рамках систем. Назовем следующие классические подходы, образующие целостные парадигмы разработки программного обеспечения:

- структурный подход, предписывающий собирать программы из процедур, взаимодействующих посредством вызовов [1];
- объектно-ориентированный подход (ООП), в рамках которого в предметной области выделяются автономные иерархически организованные сущности (объекты), взаимодействующие путем обмена сообщениями [2];
- аспектно-ориентированный подход (АОП), когда система собирается из программных реализаций классов задач (concerns), самостоятельно определяющих условия и содержание взаимодействия со своим окружением [3].

При создании крупных многоцелевых систем, пересекающих границы нескольких предметных областей, целесообразно использовать различные подходы для синтеза различных подсистем. Предлагаются универсальные методологии сборки систем из таких разнородных составляющих, например проектирование, управляемое моделями (model

ISSN 1818-7897. Вестник НГУ. Серия: Математика, механика, информатика. 2012. Т. 12, вып. 3. С. 103–126

driven architecture, MDA) [4]. Здесь требуется описывать акты комплексирования и анализировать свойства их результатов в абстрактных терминах теории систем, не зависящих от конкретной парадигмы декомпозиции.

Существует математический аппарат, специально приспособленный для формального описания такого рода, — теория категорий [5]. Она предназначена для характеризации (математических) объектов свойствами их взаимосвязей с другими объектами (морфизмов), без необходимости рассматривать внутреннюю структуру объектов (принцип «черного ящика»). Индивидуальные объекты идентифицируются по экстенсиональным («общесистемным») критериям, таким как универсальность (существование и единственность связи с аналогичными объектами), естественность (независимость многошаговой связи от способа ее прослеживания — иначе говоря, коммутативность) и т. д. Обычно такие объекты задаются с точностью до изоморфизма — формального представления несущественного различия. Для наглядной записи критериев идентификации объектов активно используется графический язык диаграмм. В настоящей работе в основном исследуется универсальная конструкция копредела диаграммы.

Представление единиц и способов комплексирования программных систем объектами и морфизмами подходящих категорий позволяет формально специфицировать и верифицировать интеграционные свойства, присущие системе как целое, а не отдельным частям. Примерами таких свойств является масштаб, связность, отказоустойчивость [6]. Процессы синтеза и анализа систем задаются как процедуры идентификации объектов в рамках формальных дисциплин (schools, см. [7]). Дисциплина — это набор взаимосвязанных категорий, состав и свойства которых отражают особенности различных видов процессов: проектирование, специфицирование, конфигурирование систем. Рассмотрению разнообразных классов дисциплин посвящена настоящая работа.

Подчеркнем, что в задачи работы не входит получение ни новых фундаментальных результатов в теории категорий, ни новых методов комплексирования программ. Дело в том, что многие актуальные проблемы синтеза систем вызваны не столько недостатком тех или других, сколько слабой совместимостью разнородных методов и технологий, а также конфликтом между общностью математических результатов и конкретностью практических задач. Поэтому в работе сделан акцент на единообразном формальном описании принципов комплексирования, в рамках которого проанализирован ряд задач, возникающих в практике синтеза систем. К их числу относятся применение шаблонов (patterns) комплексирования, выявление (mining) оптимальной архитектуры, выбор интеграционных интерфейсов, распараллеливание, покомпонентная трансформация (refinement) систем, разработка специализированных технологий комплексирования систем.

1. Приемы комплексирования программных систем

Способы сборки программ из небольших модулей, разработка которых может вестись независимо, относятся к ключевым возможностям любой технологии программирования. В классическом структурном программировании модуль представляет собой

набор процедур, а его интеграция в систему сводится к их вызову из других модулей, т. е. к подстановке их программных реализаций вместо имен в ходе исполнения. Чтобы интегрировать модуль, нужно изменить систему (добавить вызов), оставляя его неизменным. В ходе развития программирования концепции процедуры и модуля претерпевали существенные изменения, однако вызов процедуры остается основным механизмом межмодульного взаимодействия. Современным подходом к модуляризации процедур, в том числе удаленных от точек вызова, считается сервисно-ориентированная архитектура (service-oriented architecture, SOA).

Полноценные системы, в дополнение к программным модулям, содержат массивы данных, которые читаются, порождаются и модифицируются в ходе исполнения. Для того чтобы интегрировать в систему массив, его необходимо загрузить — преобразовать к виду, доступному для модулей. При этом модули менять не нужно. Например, для упрощения загрузки данных из реляционных баз в объектные приложения разрабатываются средства объектно-реляционного отображения (object-relational mapping, ORM).

При создании крупных систем в качестве компонентов применяются не только модули, но и готовые автономные приложения. Они недоступны для изменения, поэтому для их интеграции требуется создавать дополнительные промежуточные компоненты, способные взаимодействовать с каждым из них, посылая им и принимая от них сообщения в подходящем формате. Такие компоненты выступают в роли «клея» (glue), соединяющего приложения в систему. Программные реализации «клея», совместимые с распространенными технологиями программирования, выпускаются в виде промежуточного программного обеспечения (middleware). Примером соединения также является связывание (weaving) – основной способ комплексирования в аспектно-ориентированном подходе, состоящий в присоединении аспектов к базовым программным модулям без изменения тех и других [8].

Таким образом, имеются три приема комплексирования: загрузка, подстановка и соединение. Ясно, что любой акт сборки системы комбинируется из них, поскольку для включения части в целое можно либо изменить часть, либо изменить целое, либо добавить «клей». В структурном программировании комплексирование ограничивается тем обстоятельством, что каждый прием может быть применен только к единице соответствующего вида. Чтобы снять это ограничение, была предложена концепция объекта – автономной сущности с неизменным интерфейсом, содержащей процедуры (методы), доступные для внешнего вызова, и данные (атрибуты). Однако обнаружилось, что такая универсальность объектов порождает слишком большую степень произвола: архитекторам трудно предугадать, какой режим совместной работы объектов будет наиболее эффективным для выполнения той или иной конкретной задачи. Решение этой проблемы (заимствованное из практики проектирования зданий) заключается в выявлении шаблонов (patterns). Шаблон – это правило комплексирования объектов, обеспечивающее оптимальное решение явно описанного класса задач проектирования. Оказывается, шаблоны естественно классифицируются по приемам комплексирования – в основополагающей книге [9] все шаблоны разделены на три класса (причем без объяснения причины):

- создающие шаблоны, порождающие объекты из исходных данных, например абстрактная фабрика (AbstractFactory);
- поведенческие шаблоны, управляющие поведением объектов путем подстановки различных реализаций методов (перегрузки), например стратегия (Strategy);
- структурные шаблоны, «склеивающие» неизменяемые объекты, например адаптер (Adapter).

Прием комплексирования	Способ	Схема	Механизм интеграции	Единица комплексирования	Пример технологии	Класс шаблонов ООП
Загрузка	Изменение части	$C \to \mathcal{S}$	Загрузка данных	Массив данных	ORM	Создающие
Подстановка	Изменение целого	$C \to \mathcal{S}' \leftarrow \mathcal{S}$	Вызов процедуры	Модуль	SOA	Поведенческие
Соединение	Добавление «клея»	$C \leftarrow \mathcal{G} \to \mathcal{S}$	Обмен сообщениями	Приложение	Middleware	Структурные

2. Категория диаграмм и оптимизация архитектуры

Мнемоническим схемам приемов комплексирования, приведенным в таблице выше, можно придать точный смысл, если привлечь аппарат теории категорий — математической дисциплины, которая «начинается с наблюдения, что многие свойства математических систем можно представить просто и единообразно посредством диаграмм» [5. С. 12]. Теоретико-категорный подход к формализации синтеза систем разрабатывается начиная с 1970-х гг. [10]. В рамках этого подхода компонентам и системам сопоставляются абстрактные объекты, а любому действию по интеграции индивидуального компонента в систему — морфизм, т. е. абстрактный аналог функции, отображающий объект-область (компонент) в объект-кообласть (систему). Подчеркнем, что для компонента C и системы S, как правило, указывается не только возможность (или невозможность) интеграции C в S, но и совокупность всех различных способов интеграции, порождающая множество морфизмов $\mathrm{Mor}(C, S)$. Композиция морфизмов отвечает многошаговым действиям, причем их результат не зависит от порядка прослеживания шагов (свойство ассоциативности). Имеются тождественные морфизмы, означающие «ничегонеделание». При описании конкретных методов программирования объектами служат формальные модели программ определенного вида: алгебраические спецификации, графовые схемы, термы лямбда-исчисления и т. д. Таким образом, получается категория, обозначаемая далее как $c\text{-}DESC$.

Наборы компонентов, из которых строятся системы, и взаимосвязи между ними представляются диаграммами этой категории. Акту сборки системы из набора Δ отвечает построение копредела — диаграммы в форме коконуса с основанием Δ, обладающего свойством универсальности (определения этих понятий ввиду их важности мы приводим ниже). Вершина копредела обозначает систему, а ребра копредельного коконуса — вхождения компонентов в нее. Примером из области классического программирования служит компоновка приложений из модулей (linking): с точки зрения компоновщика, модуль представляет собой множество имен внешних процедур, а его интеграция — привязку к вызывающим их модулям путем отождествления процедур с совпадающими

именами. В простейшем случае, когда компоненты никак не взаимосвязаны (например, при объединении модулей в библиотеку), система представляет собой их прямую сумму, явным образом сохраняющую идентичность каждого компонента и не добавляющую ничего лишнего. Даже пустая диаграмма может иметь копредел: его можно создать, вообще не имея компонентов. Примером служит инициализационный массив данных, загружаемый в любой модуль единственным образом.

Схемы приемов комплексирования представляют собой схемы диаграмм. Любая диаграмма встраивания имеет копредел с вершиной S, а любая диаграмма подстановки – с вершиной S'. Копредел диаграммы соединения называется кодекартовым квадратом, он существует не всегда. Поэтому большое практическое значение имеют технологии комплексирования систем посредством соединения, гарантирующие получение адекватных результатов. Они активно разрабатываются в рамках компонентного подхода к проектированию [11].

Диаграммы категории $c\text{-}DESC$, копределы которых действительно отвечают актам сборки систем, называются конфигурациями и образуют класс, обозначаемый как $Conf$. В работе [7] класс конфигураций задается в форме отображения, сопоставляющего каждому набору объектов CD множество диаграмм $Conf(CD)$, семейство объектов каждой из которых совпадает с CD. Наш класс $Conf$ представляет собой в точности объединение всех множеств вида $Conf(CD)$.

Чтобы формализовать многошаговые процессы синтеза систем, в ходе которых одни диаграммы преобразуются в другие, класс конфигураций сам рассматривается как категория (ковариантная категория «сверхзапятой» [5]). В дальнейшем мы будем даже строить диаграммы, состоящие из диаграмм. Несложным примером приложения этих конструкций служит задача выявления архитектуры (architecture mining) системы S – нахождение ее конфигурации, оптимальной с точки зрения некоторого метода архитектурной декомпозиции. Если метод формализуется правилом построения некоторого подкласса Cd в $Conf$, то оптимальной («минимальной»/«максимальной») конфигурацией будет универсальный (инициальный/терминальный) объект в категории, состоящей из всех диаграмм из класса Cd, имеющих копредел с вершиной S. Действительно, инициальная конфигурация, единственным образом входящая в любую другую, представляет собой апробацию (proof of concept) метода, а терминальная, единственным образом поглощающая в себя любую другую, — наилучшее приближение (best approximation), получаемое с его помощью.

Перейдем к точным определениям. Мы используем (элементарные) теоретико-категорные понятия, определения которых можно найти в книгах [5; 12]. В частности, в [12] объекты (морфизмы) категории C кратко называются C-объектами (C-морфизмами), а функтор вида $\Delta : X \to C$ — C-диаграммой со схемой X. Диаграмма часто изображается как граф категории X, вершины которого помечены C-объектами, а ребра — C-морфизмами. Мы будем предполагать, что X — малая категория. Категория всех малых категорий и всех их функторов обозначается через **Cat**, она является полной подкатегорией в категории **CAT** всех категорий и всех функторов. (На самом деле **CAT** настолько «велика», что в книге [12] она названа квазикатегорией, но для целей

настоящей работы это не существенно). Диаграмма называется конечной, если таковой является ее схема.

Класс всех C-диаграмм служит классом объектов категории, конструкция которой обобщает конструкцию категории запятой (comma category, см. [5. С. 58]). За основу возьмем категорию запятой вида $D \downarrow S$, где D — подкатегория в некоторой категории E; S — произвольный E-объект. Напомним, что класс ее объектов состоит из всех E-морфизмов, направленных из D-объектов в S, а морфизмом объекта $f : X \to S$ в $g : Y \to S$ служит любой D-морфизм $d : X \to Y$, образующий коммутативный E-треугольник $f = g \circ d$. Имеется функтор $dom : (D \downarrow S) \to D : f \mapsto \mathrm{dom}\, f, d \mapsto d$, «забывающий» про S. C-диаграммы рассматриваются как объекты категории запятой $\mathbf{Cat} \downarrow C$, однако понятие морфизма для диаграмм расширяется: коммутативность треугольника допускается с точностью до естественного преобразования. Напомним, что естественным преобразованием ε функтора $fun : F \to G$ в $fun'/ : F \to G$ называется семейство G-морфизмов $\{\varepsilon_A : fun(A) \to fun'(A) \mid A \in \mathrm{Ob}\, F\}$ такое, что для любого F-морфизма $f : A \to B$ выполняется соотношение $\varepsilon_B \circ fun(f) = fun'(f) \circ \varepsilon_A$. 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 $\varepsilon : \Delta \to \Xi \circ fd$; закон композиции имеет вид $\langle \varepsilon, fd \rangle \circ \langle \gamma, gd \rangle = \langle \varepsilon_{gd} \circ \gamma, fd \circ gd \rangle$. Категория диаграмм, определенная таким способом, обозначается через $\mathbf{D}C$. Имеется функтор $sch : \mathbf{D}C \to \to \mathbf{Cat} : \Delta \mapsto \mathrm{dom}\, \Delta, \langle \varepsilon, fd \rangle \mapsto fd$. 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 имеется эндофунктор $\mathbf{D} : \mathbf{CAT} \to \mathbf{CAT} : C \mapsto \mathbf{D}C, fun \mapsto (fun \circ -)$.

Имеется вложение $ic : (\mathbf{Cat} \downarrow C) \to \mathbf{D}C$, переводящее функтор fd, удовлетворяющий условию $\Delta = \Xi \circ fd$, где Δ, Ξ – C-диаграммы, в $\mathbf{D}C$-морфизм $\langle 1_\Delta, fd \rangle$. При этом если функтор fd является вложением (т. е. инъекцией), то Δ называется поддиаграммой в Ξ (ввиду того, что граф Δ является подграфом графа Ξ). Также в $\mathbf{D}C$ вкладывается любая категория вида C^X, состоящая из всех функторов, действующих из категории X в C, и всех их естественных преобразований: естественному преобразованию ε соответствует $\mathbf{D}C$-морфизм $\langle \varepsilon, 1_X \rangle$. Категории функторов вместе с $\mathbf{Cat} \downarrow C$ порождают $\mathbf{D}C$, поскольку любой $\mathbf{D}C$-морфизм допускает разложение

$$\langle \varepsilon, fd \rangle = ic(fd) \circ \langle \varepsilon, 1_{sch(\Delta)} \rangle : \Delta \to \Xi \circ fd \to \Xi.$$

Поскольку $C^{\mathbf{1}} \cong C$, где $\mathbf{1}$ — терминальная категория, состоящая из одного объекта 0 и морфизма 1_0, то имеется (полное) вложение C в $\mathbf{D}C$, которое мы будем обозначать через $\ulcorner - \urcorner$. Диаграммы вида $\ulcorner S \urcorner$, где S – произвольный C-объект, мы будем называть точками. Определим конструкцию коконуса, играющую ключевую роль в формальном анализе комплексирования систем.

Определение 1. *Коконусом* называется $\mathbf{D}C$-морфизм, имеющий точку в качестве кообласти.

Зафиксируем произвольную диаграмму $\Delta : X \to C$. В литературе коконус с областью Δ часто определяют как естественное преобразование из Δ в постоянный функтор, действующий из X в C (см., например, [5. С. 81–82]). Это определение эквивалентно

нашему во всех случаях, за исключением того, когда X — пустая категория. В этом последнем случае существует в точности один функтор, действующий из X в C (это пустая диаграмма), поэтому нельзя определить инициальный объект как вершину универсального коконуса с пустой диаграммой в качестве области.

Область Δ коконуса $\langle \sigma, !_X : X \to \mathbf{1} \rangle : \Delta \to \ulcorner S \urcorner$ называется его основанием, объект S — вершиной, компоненты естественного преобразования σ — ребрами. Коконус можно рассматривать как объект категории запятой $(\mathbf{D}C) \downarrow \ulcorner S \urcorner$. В то же время он является диаграммой со схемой X в категории запятой $C \downarrow S$, получающейся из Δ путем замены объектов исходящими из них ребрами. Эти представления коконусов эквивалентны:

Предложение 1. $((\mathbf{D}C) \downarrow \ulcorner S \urcorner) \cong \mathbf{D}(C \downarrow S)$ *для любого* C*-объекта* S.

ДОКАЗАТЕЛЬСТВО. Любой коконус $\langle \sigma, !_X \rangle : \Delta \to \ulcorner S \urcorner$ взаимно-однозначно определяет $(C \downarrow S)$-диаграмму, переводящую любой X-объект I в σ_I и любой X-морфизм f в $\Delta(f)$. □

Пользуясь этим фактом, легко определить понятие *подкоконуса*: коконус θ является подкоконусом коконуса δ с вершиной S, если θ является поддиаграммой δ в $\mathbf{D}(C \downarrow S)$.

Копределом диаграммы Δ называется коконус $\operatorname{colim} \Delta : \Delta \to \ulcorner S \urcorner$, универсальный в том смысле, что для любых C-объекта T и коконуса $\delta : \Delta \to \ulcorner T \urcorner$ существует единственный C-морфизм $u : S \to T$ такой, что $\delta = \ulcorner u \urcorner \circ \operatorname{colim} \Delta$. При этом S называется объектом копредела, а u – стрелкой копредела. Копредел определяется однозначно с точностью до изоморфизма: коконус δ является копределом диаграммы Δ тогда и только тогда, когда существует C-изоморфизм i такой, что $\delta = \ulcorner i \urcorner \circ \operatorname{colim} \Delta$. В свою очередь, если $\mu : \Xi \to \Delta$ — $\mathbf{D}C$-изоморфизм, то $(\operatorname{colim} \Delta) \circ \mu$ — копредел диаграммы Δ.

Если X — пустая категория, то объект копредела представляет собой в точности инициальный C-объект. Если X — дискретная категория (т. е. не имеющая нетождественных морфизмов), то объект копредела называется копроизведением семейства объектов $\Delta(\operatorname{Ob} X)$. Напомним в связи с этим, что все дискретные категории образуют полную корефлексивную подкатегорию в **CAT**, которую мы обозначим через **d-CAT**: имеется канонический функтор $|-| : \mathbf{CAT} \hookrightarrow \mathbf{d\text{-}CAT}$, «забывающий» все нетождественные морфизмы, и его естественное преобразование ι в тождественный функтор $1_{\mathbf{CAT}}$, состоящее из вложений, тождественных на объектах. Функтор $|-|$ переводит диаграммы в свои поддиаграммы: соотношение $\iota_C \circ |\Delta| = \Delta \circ \iota_X$ определяет $(\mathbf{Cat} \downarrow C)$-морфизм $\iota_X : \iota_C \circ |\Delta| \to \Delta$, порождающий вложение диаграмм $\iota_\Delta = ic(\iota_X)$.

Нам потребуется рассматривать различные функторы с точки зрения их воздействия на копределы. Согласно [12. Разд. 13], говорится, что функтор $fun : C \to D$:

- *сохраняет* копределы диаграммы Δ, если для любого копредела $\delta : \Delta \to \ulcorner S \urcorner$ коконус $fun \circ \delta : fun \circ \Delta \to \ulcorner fun(S) \urcorner$ является копределом диаграммы $fun \circ \Delta$;
- *поднимает* копределы диаграммы Δ, если для любого копредела $\xi : fun \circ \Delta \to \ulcorner T \urcorner$ существует копредел $\delta : \Delta \to \ulcorner S \urcorner$ такой, что $fun \circ \delta = \xi$.

Пусть Cd – класс C-диаграмм, имеющих копределы. Можно рассматривать его как полную подкатегорию в $\mathbf{D}C$ и определить функтор копредела $colim$, действующий из него

в C, сопоставляя каждой диаграмме из Cd объект некоторого ее копредела, а каждому $\mathbf{D}C$-морфизму $\theta : \Delta \to \Xi$, где $\Delta, \Xi \in Cd$ — стрелку копредела $colim(\theta)$ такую, что $\operatorname{colim}\Xi \circ \theta = \ulcorner colim(\theta) \urcorner \circ \operatorname{colim}\Delta$.

$$\begin{array}{ccc} \Delta & \xrightarrow{\operatorname{colim}\Delta} & \ulcorner colim(\Delta)\urcorner \\ {\scriptstyle\theta}\downarrow & & \downarrow{\scriptstyle\ulcorner colim(\theta)\urcorner} \\ \Xi & \xrightarrow{\operatorname{colim}\Xi} & \ulcorner colim(\Xi)\urcorner \end{array}$$

Чтобы выделить в Cd диаграммы с фиксированным объектом копредела, будем обозначать через $Cd \downarrow S$ полную подкатегорию в $(\mathbf{D}C) \downarrow \ulcorner S \urcorner$, объектами которой являются все копределы диаграмм из Cd, имеющие вершину S. Эта конструкция используется для формализации задачи оптимизации архитектуры.

Определение 2. Класс Cd C-диаграмм, имеющих копределы, называется *оптимизируемым*, если любая категория $Cd \downarrow S$ обладает инициальным и терминальным объектами.

Теперь кратко рассмотрим диаграммы, составленные из диаграмм, т. е. перейдем в категорию $\mathbf{DD}C$. Поскольку **Cat** кополна [13], $\mathbf{D}C$ имеет некоторые копределы (в частности, инициальный объект и копроизведения) независимо от существования каких-либо копределов в C. Это вытекает из следующего утверждения.

Предложение 2. $\mathbf{D}C$*-диаграмма* $ic \circ \Gamma$ *имеет копредел для любой* $(\mathbf{Cat} \downarrow C)$*-диаграммы* Γ.

ДОКАЗАТЕЛЬСТВО. Копредел диаграммы $\Gamma : X \to (\mathbf{Cat} \downarrow C)$ строится, как в любой категории запятой: он имеет вид $\operatorname{colim}\Gamma'^{/}\colon \Gamma \to \ulcorner \Pi \urcorner$, где $\Gamma'^{/} = dom \circ \Gamma : X \to \mathbf{Cat}$, $\Pi : colim(\Gamma') \to C$ — единственный функтор, удовлетворяющий условию $\delta = \ulcorner \Pi \urcorner \circ \operatorname{colim}\Gamma'^{/}$, $\delta : \Gamma'^{/} \to \ulcorner C \urcorner$ — коконус, изоморфный Γ в смысле предложения 1. Остается заметить, что функтор ic сохраняет копределы: в этом нетрудно убедиться, используя явную конструкцию копредела в **Cat**, описанную в [13]. □

Используя копределы в $\mathbf{Cat} \downarrow C$, можно построить естественную проекцию (функтор отрисовки) $\mathbf{K} : \mathbf{DD}C \to \mathbf{D}C$. Грубо говоря, отрисовка диаграммы $\Gamma : X \to \mathbf{D}C$ заключается в замене каждой вершины A графа категории X графом C-диаграммы $\Gamma(A)$, а каждого ребра $f : A \to B$ — совокупностью ребер, по одному для каждой вершины I графа диаграммы $\Gamma(A)$, направленному из I в вершину $fd(I)$ графа $\Gamma(B)$ и помеченному C-морфизмом ε_I, где $\langle \varepsilon, fd \rangle = \Gamma(f)$. Читателей, искушенных в теории категорий, отошлем за строгим определением отрисовки к работе [14]: в ней показано, что тройка $\langle \mathbf{D}, |-|, \mathbf{K} \rangle$ образует монаду в **CAT**, по своим свойствам аналогичную монаде степени в категории **Set** всех множеств и всех отображений, имеющей вид $\langle 2^{-}\!/\{-\}, \bigcup\!\!\bigwedge \rangle$.

Отрисовка позволяет рассматривать коконусы как диаграммы. Будем обозначать через $\llcorner - \lrcorner$ вложение категории морфизмов $C^{\mathbf{2}}$ в $\mathbf{D}C$ (напомним, что символом **2** обозначается двухэлементное линейно упорядоченное множество $\{0 \to 1\}$, рассматриваемое как категория, так что диаграммы вида $\mathbf{2} \to C$ — это в точности все C-морфизмы). Будем говорить, что C-диаграмма *имеет форму коконуса*, если она имеет вид $\mathbf{K}\llcorner \delta \lrcorner$ для

некоторого коконуса δ (здесь $\llcorner - \lrcorner : (\mathbf{D}C)^{\mathbf{2}} \hookrightarrow \mathbf{DD}C$). Обозначим через $\mathrm{Cocone}\, C$ класс всех C-коконусов.

Предложение 3. *Класс* $\mathrm{Cocone}\, C$ *оптимизируем.*

ДОКАЗАТЕЛЬСТВО. Копредел диаграммы, имеющей форму коконуса $\delta : \Delta \to \ulcorner S \urcorner$, имеет вид $\mathbf{K}(\langle \bar{\delta}, !_{\mathbf{2}} \rangle : \llcorner \delta \lrcorner \to \ulcorner\ulcorner S \urcorner\urcorner)$, где $\bar{\delta} : \delta \to \ulcorner 1_S \urcorner \in \mathrm{Mor}\,(\mathbf{D}C)^{\mathbf{2}}$ — естественное преобразование с компонентами $\bar{\delta}_0 = \delta$ и $\bar{\delta}_1 = \ulcorner 1_S \urcorner$. Инициальным объектом в категории $(\mathrm{Cocone}\, C) \downarrow S$ является копредел диаграммы $\ulcorner S \urcorner$ (отрисовки коконуса, направленного из пустой $\mathbf{D}C$-диаграммы в $\ulcorner S \urcorner$), а терминальным — $\mathrm{colim} \llcorner 1_S \lrcorner$ (здесь снова $\llcorner - \lrcorner : C^{\mathbf{2}} \hookrightarrow \mathbf{D}C$). □

3. Формальные дисциплины проектирования

Чтобы полностью описать процесс синтеза программных систем, недостаточно рассмотреть только конфигурации: необходимо также ввести понятия интерфейса и трансформации. В частности, хорошо известно, что интеграционные возможности компонентов определяются их интерфейсами — специально подготовленными частями, видимыми «снаружи». Примером интерфейса служит сигнатура программного модуля — список процедур, реализованных в нем, с указанием параметров и возвращаемых значений. Формальные модели интерфейсов образуют категорию, обозначаемую через SIG, а операция выделения интерфейса формализуется как функтор $sig : c\text{-}DESC \to SIG$, называемый сигнатурным. Поскольку различные компоненты могут иметь один и тот же интерфейс, sig не обязан быть инъективным на объектах. Однако sig-образы двух различных действий, интегрирующих один и тот же компонент в одну и ту же систему, должны быть различными: иначе получится, что интерфейсы недостаточно детально описывают интеграционные возможности компонентов. Иными словами, функтор sig должен быть унивалентным (faithful), т. е. инъективным на каждом множестве вида $\mathrm{Mor}(A, B)$, состоящем из всех морфизмов с областью A и кообластью B.

Для каждого интерфейса гарантируется наличие хотя бы одной реализации, поддерживающей его интеграционные возможности в полном объеме. Она называется (sig-) дискретной и задается функтором $sig^{*\diagup} : SIG \to c\text{-}DESC$. Интерфейс дискретной реализации любого SIG-объекта I должен совпадать с I, поэтому должно выполняться соотношение $sig \circ sig^{*\diagup} = 1_{SIG}$. Полнота дискретной реализации заключается в том, что для любого SIG-объекта I и $c\text{-}DESC$-объекта S функтор sig сюръективно (следовательно, биективно) отображает множество $\mathrm{Mor}(sig^*(I), S)$, описывающее все действия по интеграции компонента $sig^*(I)$ в систему S, на множество $\mathrm{Mor}(I, sig(S))$, определяющее интеграционные возможности интерфейса I. Эти условия эквивалентны тому, что функтор $sig^{*\diagup}$должен быть сопряженным слева к sig с тождественной единицей. В частности, он задает полное вложение категории SIG в $c\text{-}DESC$. Например, дискретные реализации сигнатур программных модулей состоят из «заглушек» (stubs) — пустых процедур; они могут автоматически генерироваться CASE-средствами и позволяют быстро собирать отладочные версии приложений.

Существует естественная взаимосвязь между интерфейсами и конфигурациями. Ес-

ли две c-$DESC$-диаграммы имеют один и тот же $\mathbf{D}sig$-образ, то они обе должны либо принадлежать классу $Conf$, либо нет. Это условие задает своего рода логический закон непротиворечия для интерфейсов: совокупность всех наборов компонентов, имеющих фиксированную схему интеграции интерфейсов, не должна содержать как конфигурации, так и наборы, не порождающие целостных систем. Кроме того, выделение интерфейса должно быть естественным относительно комплексирования (т. е. коммутативным с ним) в том смысле, что копредел $\mathbf{D}sig$-образа любой конфигурации должен быть $\mathbf{D}sig$-образом ее копредела (т. е. sig поднимает копределы конфигураций). Как мы увидим далее (предложение 5), из этого и других указанных выше свойств функтора sig вытекает естественность в обратную сторону: sig сохраняет копределы конфигураций.

Полноценный процесс создания систем в дополнение к актам интеграции содержит трансформации (refinements) – шаги разработки индивидуальных компонентов (уточнение требований, реализация спецификации на языке программирования и др.). Процесс рассматривается как идущий в двух измерениях: по горизонтали располагаются действия по интеграции, реализующие отношения «часть-целое», а по вертикали – трансформации, реализующие отношения «абстрактное-конкретное» [15]. Класс всех трансформаций задается как категория, обозначаемая r-$DESC$, она обладает теми же объектами, что и c-$DESC$, но другими морфизмами. Тривиальным частным случаем трансформации является c-$DESC$-изоморфизм: единицы комплексирования, несущественно отличающиеся друг от друга с точки зрения отношения «часть-целое», не должны отличаться и по отношению «абстрактное-конкретное» [7].

Условие естественности трансформаций относительно комплексирования систем налагается в следующей форме: любой набор трансформаций объектов некоторой конфигурации должен порождать трансформацию собираемой из них системы как целого. Это условие можно формализовать с привлечением естественных преобразований, состоящих из r-$DESC$-морфизмов, поскольку любая дискретная c-$DESC$-диаграмма является также r-$DESC$-диаграммой. Произвольный набор трансформаций объектов диаграммы $\Delta : X \to c\text{-}DESC$ представляет собой в точности естественное преобразование $\varphi : |\Delta| \to \Sigma \in \operatorname{Mor} r\text{-}DESC^{|X|}$/дискретной диаграммы $|\Delta|$ в дискретную диаграмму Σ, состоящую из результатов всех трансформаций. Если $\Delta \in Conf$, то Σ должна быть поддиаграммой некоторой конфигурации, объект копредела которой может быть получен путем трансформации из $colim(\Delta)$.

Изложенные соображения приводят к следующей сложной формальной конструкции, в рамках которой можно проводить всесторонний анализ процессов синтеза программных систем. Она почти полностью совпадает с предложенной в [7].

Определение 3. *Формальной дисциплиной проектирования* (architecture school) называется четверка $\langle c\text{-}DESC, Conf, sig, r\text{-}DESC\rangle$, где:

• c-$DESC$ — категория, объекты которой называются описаниями (descriptions) или формальными моделями компонентов, а морфизмы называются действиями по интеграции (компонентов в системы);

• $Conf$ — класс c-$DESC$-диаграмм, называющихся допустимыми конфигурациями систем (configurations);

• sig — функтор из $c\text{-}DESC$ в категорию, обозначаемую SIG, объекты которой называются интерфейсами или сигнатурами компонентов (signatures), а морфизмы называются действиями по интеграции интерфейсов;

• $r\text{-}DESC$ – категория, объектами которой являются формальные модели, а морфизмы называются трансформациями (refinements) 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 $c\text{-}DESC$-диаграмм Δ, Ξ, если $sig \circ \Delta = sig \circ \Xi$ и $\Delta \in Conf$, то $\Xi \in Conf$.

(vi) $\mathrm{Ob}\, r\text{-}DESC = \mathrm{Ob}\, c\text{-}DESC$.

(vii) Подкатегория в $c\text{-}DESC$, состоящая из всех $c\text{-}DESC$-объектов и всех изоморфизмов, является подкатегорией в $r\text{-}DESC$.

(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 Δ в качестве поддиаграммы, и $r\text{-}DESC$-морфизм $r : colim(\Delta) \to colim(\Delta \oplus \varphi)$.

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

Произвольная категория C порождает тривиальную формальную дисциплину проектирования $\langle C, \emptyset, 1_C, (\mathrm{Ob}\, C, \mathrm{Iso}\, C)\rangle$. Нетривиальные примеры дисциплин приведены в работах [7; 16; 17]. Формализация многих известных методов моделирования программ приводят к дисциплинам «над» категорией **Set**. Моделями в такой дисциплине служат множества, на которых задана некоторая структура (алгебраические системы, топологические пространства, графы и т. п.), а действиями по интеграции — отображения множеств, совместимые со структурой (в подходящем смысле); sig является каноническим функтором взятия основного множества, «забывающим» структуру, а левый сопряженный к нему порождает на произвольном множестве минимальную (дискретную) структуру. В качестве трансформаций обычно выступают антифункции — отношения, задающие «расширение» точек основного множества трансформируемых моделей до подмножеств. Мы подробно рассмотрим их в конце разд. 5.

Характерным примером служит дисциплина моделирования сценариев поведения систем множествами событий, частично упорядоченными причинно-следственной связью [17]. В конфигурациях сценариев результат комплексирования присутствует явно (с точностью до раздельного объединения). Трансформация сценария X в Y – это отношение $R \in X \times Y$, удовлетворяющее следующим условиям:

• $\forall y \in Y\ \exists! x \in X\ xRy$ (антифункциональность);

• $\forall x \in X\ \exists y \in Y\ xRy$ (тотальность);

• $\forall x, x'^{/} \in X\ \forall y, y'^{/} \in Y\ ((xRy \wedge x'Ry'^{/} \wedge x \neq x') \Rightarrow (x \quad x'^{/} \Leftrightarrow y \quad y'))$.

Рассмотрим четверку

$$SM = \langle \mathbf{Pos}, CPos, |-|, r\text{-}Pos \rangle,$$

где **Pos** — категория всех частично упорядоченных множеств и всех их гомоморфизмов; $CPos$ — класс всех копроизведений **Pos**-коконусов; $|-| : \mathbf{Pos} \to \mathbf{Set}$ – канонический функтор, забывающий порядок; $r\text{-}Pos$ — категория, состоящая из всех частично упорядоченных множеств и всех трансформаций сценариев. Покажем, что она является формальной дисциплиной проектирования. Действительно, условие (iii) определения 3 обеспечивается функтором дискретного упорядочения множеств $id : \mathbf{Set} \to \mathbf{Pos} : S \mapsto \langle S, = \rangle$. В силу предложения 6 (см. ниже) тройка $SS = \langle \mathbf{Pos}, CPos, |-| \rangle$ является формальной дисциплиной специфицирования. Далее, любой **Pos**-изоморфизм задает трансформацию сценариев. Справедливость условия (viii) вытекает из предложения 14, доказанного в разд. 5.

Вернемся к рассмотрению произвольных дисциплин. Докажем, что выделение интерфейсов перестановочно с комплексированием. Предварительно докажем одно общее теоретико-категорное утверждение. Будем говорить, что функтор *детерминирует* копределы диаграммы Δ, имеющей копредел, если он сохраняет и поднимает их.

Предложение 4. *Функтор fun детерминирует копределы диаграммы Δ тогда и только тогда, когда он поднимает их и диаграмма $fun \circ \Delta$ имеет копредел.*

ДОКАЗАТЕЛЬСТВО. Необходимость очевидна, докажем достаточность. Пусть δ — произвольный копредел диаграммы Δ, ξ — произвольный копредел диаграммы $fun \circ \Delta$. По условию Δ имеет копредел $\delta'^{/}$такой, что $fun \circ \delta'^{/} = \xi$. Имеем $\delta = \ulcorner i \urcorner \circ \delta'^{/}$для некоторого изоморфизма i, поэтому $fun \circ \delta = \ulcorner fun(i) \urcorner \circ \xi$ является копределом диаграммы $fun \circ \Delta$. □

Зафиксируем произвольно категорию $c\text{-}DESC$ и тройку $SC = \langle c\text{-}DESC, Conf \subseteq \subseteq \mathrm{Ob}\,\mathbf{D}c\text{-}DESC, sig : c\text{-}DESC \to SIG \rangle$.

Предложение 5. *Если SC является дисциплиной специфицирования, то функтор sig сохраняет копределы конфигураций.*

ДОКАЗАТЕЛЬСТВО. Пусть η, ε — единица и коединица сопряжения $sig^{*/} \dashv sig$; напомним, что η — естественное преобразование функтора 1_{SIG} в $sig \circ sig^{*/} = 1_{SIG}$, причем оно состоит из тождественных морфизмов, а ε — естественное преобразование функтора $sig^{*/} \circ sig$ в $1_{c\text{-}DESC}$. Согласно стандартному определению коединицы через единицу (см. [5, стр. 99]) $sig(\varepsilon_S) \circ \eta_{sig(S)} = 1_{sig(S)}$ для любого $S \in \mathrm{Ob}\, c\text{-}DESC$, откуда $sig(\varepsilon_S) = 1_{sig(S)}$. Из унивалентности функтора sig вытекает, что ε_S — эпимономорфизм.

Выберем диаграмму $\Delta \in Conf$. В силу закона непротиворечия имеем $\Delta^{*/} = sig^{*/} \circ sig \circ \Delta \in Conf$, поэтому существует копредел $\sigma : \Delta^{*/} \to \ulcorner S \urcorner$. Рассмотрим коконус $\sigma^{*/} = sig^{*/} \circ sig \circ \sigma : \Delta^{*/} \to \ulcorner S^* \urcorner$, где $S^{*/} = sig^*(sig(S))$. Пусть $u : S \to S^{*/}$— стрелка копредела такая, что $\sigma^{*/} = \ulcorner u \urcorner \circ \sigma$. Поскольку $\sigma = \ulcorner \varepsilon_S \urcorner \circ \sigma^*$, имеем $\varepsilon_S \circ u = 1_S$, так что ε_S, будучи обратимым справа мономорфизмом, является изоморфизмом, следовательно, $\sigma^{*/}$— копредел диаграммы Δ^*. Отсюда, ввиду унивалентности функтора $sig^{*/}$, $sig \circ \sigma^{*/} = sig \circ \sigma$ является копределом диаграммы $sig \circ \Delta^{*/} = sig \circ \Delta$. В силу предложения 4 sig сохраняет копределы Δ. □

Этот факт позволяет выявить все *c-DESC*-диаграммы, способные служить конфигурациями. Обозначим через $ICMax_{sig}$ класс всех *SIG*-диаграмм Θ таких, что Θ имеет копредел и *sig* поднимает копределы всех диаграмм из класса $\mathbf{D}sig^{-1}(\Theta) = \{\Delta \mid sig \circ \Delta = \Theta\}$.

Предложение 6. *Тройка SC является дисциплиной специфицирования тогда и только тогда, когда выполнены условия (ii) и (iii) определения 3, и $Conf = \mathbf{D}sig^{-1}(IC)$ для некоторого класса SIG-диаграмм $IC \subseteq ICMax_{sig}$.*

ДОКАЗАТЕЛЬСТВО. Необходимость вытекает из предложения 5 при выборе $IC = sig \circ Conf$. Достаточность проверяется непосредственно. □

Рассмотрим два «предельных» случая выбора интерфейсов: когда они совпадают с моделями (с точностью до изоморфизма), так что дисциплина никак не отражает реализацию компонентов, и когда все модели имеют один и тот же интерфейс (образующий сингулярную категорию, изоморфную терминальной категории **1**).

Предложение 7. *Если функтор sig является изоморфизмом, то SC является дисциплиной специфицирования тогда и только тогда, когда пара $\langle c\text{-}DESC, Conf\rangle$ является дисциплиной конфигурирования.*

ДОКАЗАТЕЛЬСТВО. Любой изоморфизм категорий *c-DESC* и *SIG* удовлетворяет условиям (ii)–(v) определения 3 (в частности, обратный к нему является левым сопряженным). □

Примером применения этого результата служит построение «дисциплины специфицирования интерфейсов» — тройки $\langle SIG, sig{\circ}Conf, 1_{SIG}\rangle$, которая ввиду предложения 5 действительно является формальной дисциплиной специфицирования, если таковой является *SC*.

Предложение 8. *Если $SIG \cong \mathbf{1}$, то SC является дисциплиной специфицирования тогда и только тогда, когда выполняются следующие условия:*

(i) *c-DESC является предпорядком, имеющим наименьший элемент;*

(ii) *множество объектов любой диаграммы $\Delta \in Conf$ имеет точную верхнюю грань;*

(iii) *если $\Delta \in Conf$, то любая c-DESC-диаграмма Ξ со схемой $sch(\Delta)$ является конфигурацией.*

ДОКАЗАТЕЛЬСТВО. (i) Функтор $!_{c\text{-}DESC} : c\text{-}DESC \to \mathbf{1}$ унивалентен тогда и только тогда, когда из любого *c-DESC*-объекта в любой другой направлено не более одного морфизма, т. е. когда *c-DESC* является предпорядком, рассматриваемым как категория. Левый сопряженный к $!_{c\text{-}DESC}$, если он существует, переводит единственный объект категории **1** в наименьший элемент этого предпорядка, единица этого сопряжения тождественна.

(ii) Копределом любой диаграммы предпорядка Δ, если он существует, является точная верхняя грань семейства $\Delta(\mathrm{Ob}\, sch(\Delta))$, и функтор $!_{c\text{-}DESC}$ поднимает его.

(iii) Условие $!_{c\text{-}DESC} \circ \Delta = !_{c\text{-}DESC} \circ \Xi$ эквивалентно условию $sch(\Delta) = sch(\Xi)$.

□

Примером дисциплины специфицирования с $SIG \cong \mathbf{1}$ служит задание спецификаций в форме множеств аксиом — предложений формального исчисления фиксированной сигнатуры, выступающей в качестве единственного интерфейса (см., например, [18]). Предпорядок определяется теоретико-множественным включением теорий, определяемых спецификациями (т. е. замыканий множеств аксиом по отношению выводимости). Наименьшая аксиоматическая спецификация имеет пустое множество аксиом, в ней истинны только тавтологии. Комплексирование сложной спецификации состоит в теоретико-множественном объединении спецификаций всех компонентов.

В заключительной части раздела заметим, что результаты актов синтеза систем определяются с точностью до изоморфизма (условия (i), (iii), (vii) определения 3), хотя действия по интеграции интерфейсов задаются однозначно (ср. условия (ii), (v)). В связи с этим представляет интерес выявить условия, при которых можно задавать и конфигурации с точностью до изоморфизма. Оказывается, критерием здесь служит распространение закона непротиворечия интерфейсов на действия изоморфизмов.

Предложение 9. *Следующие условия эквивалентны в любой дисциплине специфицирования SC:*

(i) *класс* $Conf$ *замкнут относительно изоморфизмов диаграмм;*

(ii) *класс* $sig \circ Conf$ *замкнут относительно изоморфизмов диаграмм;*

(iii) *для любой пары c-DESC-диаграмм* Δ, Ξ*, если* $sig \circ \Delta \cong sig \circ \Xi$ *и* $\Delta \in Conf$*, то* $\Xi \in Conf$*.*

ДОКАЗАТЕЛЬСТВО. (i) $\Rightarrow$ (ii) Выберем диаграммы $\Delta \in Conf$ и $\Theta \cong sig \circ \Delta$. Имеем $sig^* \diagup \Theta \cong sig^* \diagup sig \circ \Delta \in Conf$, откуда $sig^* \diagup \Theta \in Conf$ по предположению, поэтому $\Theta = sig \circ (sig^* \diagup \Theta) \in sig \circ Conf$.

(ii) $\Rightarrow$ (iii) Пусть Δ, Ξ — c-$DESC$-диаграммы такие, что $sig \circ \Delta \cong sig \circ \Xi$ и $\Delta \in Conf$. По предположению, существует конфигурация $\Xi' \diagup$такая, что $sig \circ \Xi = sig \circ \Xi'$, откуда $\Xi \in Conf$ ввиду условия (v) определения 3.

(iii) $\Rightarrow$ (i) Очевидно. □

4. Распараллеливание

Интерфейсы играют важную роль в решении задачи распараллеливания — разбиения некоторой системы на части, в той или иной степени изолированные друг от друга. Это частный случай задачи выявления архитектуры, поэтому с формальной точки зрения она ставится в классе коконусов определенного вида. Действительно, степень изоляции компонентов можно формально оценить как степень структурного сходства модели системы с копроизведением моделей своих компонентов. Конечно, 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.

Зафиксируем произвольную формальную дисциплину специфицирования $SC = \langle c\text{-}DESC, Conf, sig\rangle$.

Определение 4. $c\text{-}DESC$-коконус $\pi : \Delta \to \ulcorner S \urcorner$ называется (sig-)разбиением модели S, если он удовлетворяет следующим условиям:

(i) диаграмма Δ дискретна;

(ii) π является подкоконусом копредела некоторой конфигурации;

(iii) $sig \circ \pi$ является копределом;

(iv) если θ — собственный подкоконус π, то $sig \circ \theta$ не является копределом.

Ребра разбиения называются его *компонентами*. Дисциплина SC называется *поддерживающей* некоторый класс разбиений, если он оптимизируем (в смысле определения 2). Разбиение называется *суммой*, если оно является копределом. Дисциплина SC называется *поддерживающей параллелизм*, если класс всех сумм оптимизируем.

Тривиальное однокомпонентное «распараллеливание» системы фактически сохраняет интерфейсы, поэтому его можно назвать перегрузкой, заимствуя термин из объектно-ориентированного программирования.

Определение 5. $c\text{-}DESC$-морфизм o называется *перегрузкой*, если порожденный им коконус $\ulcorner o \urcorner$ является разбиением.

Покажем, что при наличии достаточного количества конфигураций среди перегрузок имеются оптимальные. Обозначим через $Ovload$ класс всех перегрузок. Заметим, что $sig(Ovload) \subseteq \operatorname{Iso} SIG$, поэтому $Ovload \subseteq \operatorname{Epi} SIG \cap \operatorname{Mono} SIG$ ввиду унивалентности функтора sig.

Предложение 10. *Если* $Conf \supseteq \{\llcorner 1_{\operatorname{codom} o} \lrcorner \mid o \in Ovload\}$, *то* SC *поддерживает перегрузки.*

ДОКАЗАТЕЛЬСТВО. Пусть, как в доказательстве предложения 5, ε — коединица сопряжения $sig^* \dashv sig$. Пусть S — кообласть некоторой перегрузки; в частности, $sig(S)$ не является инициальным SIG-объектом (в противном случае единственное разбиение S имеет пустое основание). Имеем $\llcorner \varepsilon_S \lrcorner \in Conf$, поскольку $sig(\varepsilon_S) = sig(1_S)$. Копредел диаграммы $\llcorner \varepsilon_S \lrcorner$ является инициальным объектом в $\llcorner Ovload \lrcorner \downarrow S$, а диаграммы $\llcorner 1_S \lrcorner$ — терминальным. В частности, для любой перегрузки $o : T \in S$ имеем $\varepsilon_S \circ sig^*(sig(o)) = o \circ \varepsilon_T$ (в силу определения коединицы), а поскольку $sig(o)$ — изоморфизм, имеется $(\llcorner Ovload \lrcorner \downarrow S)$-морфизм $\llcorner \varepsilon_T \circ sig^*(sig(o))^{-1} \lrcorner : \llcorner \varepsilon_S \lrcorner \to \llcorner o \lrcorner$. □

Предложение 11. *Перегрузка* o *порождает сумму тогда и только тогда, когда она является изоморфизмом.*

Предположим, что SC является дисциплиной над $\mathbf{Set}$. Тогда $|S| = \coprod\!\!\bigwedge_{\in \operatorname{Ob} sch(\Delta)} |\Delta(I)|$ для любого разбиения $\pi : \Delta \to \ulcorner S \urcorner$, т. е. основные множества компонентов образуют разбиение (в буквальном смысле этого слова) основного множества системы. Если π не является суммой, то S обладает структурой, дополнительной по отношению к структурам всех $\Delta(I)$. Любая перегрузка представляет собой биекцию, поэтому она сводится к обогащению структуры. Если структура позволяет ввести понятие связности (и имеется

достаточно дискретных конфигураций), то SC поддерживает параллелизм: инициальным объектом в $Sums \downarrow S$, где $Sums$ — класс всех сумм, является копредел разбиения на связные компоненты, а терминальным — $\mathrm{colim}\llcorner 1_S \lrcorner$. Примером служит дисциплина моделирования сценариев SM.

5. Трансформации конфигураций

Зафиксируем произвольную формальную дисциплину проектирования

$$AR = \langle c\text{-}DESC, Conf, sig, r\text{-}DESC \rangle.$$

Условие (viii) определения 3 можно наглядно изобразить в виде следующего помеченного графа, отвечающего трансформации конфигураций:

$$\begin{array}{ccccc} |\Delta| & \longrightarrow & \Delta & \xrightarrow{\mathrm{colim}\,\Delta} & \ulcorner colim(\Delta)\urcorner \\ {\scriptstyle \langle \varphi, 1_{|sch(\Delta)|} \rangle}\downarrow & & & & \downarrow{\scriptstyle \ulcorner r \urcorner} \\ \Sigma & \longrightarrow & \Delta \oplus \varphi & \xrightarrow{\mathrm{colim}\,\Delta\oplus\varphi} & \ulcorner colim(\Delta \oplus \varphi)\urcorner \end{array}$$

Конечно, этот граф не изображает диаграмму, поскольку его горизонтальные ребра помечены $\mathbf{D}c\text{-}DESC$-морфизмами, а вертикальные — $\mathbf{D}r\text{-}DESC$-морфизмами (здесь напомним, что условие (vii) определения 3 позволяет рассматривать дискретные $c\text{-}DESC$-диаграммы как $r\text{-}DESC$-диаграммы). Поэтому при анализе трансформаций систем в общем случае не удается применить традиционные теоретико-категорные методы, состоящие в выявлении условий естественности, универсальности и т. д. Чтобы воспользоваться ими, необходимо перевести все составляющие графа трансформации конфигураций в категорию диаграмм подходящей категории. Простой пример такого рода получается, если Δ и $\Delta \oplus \varphi$ представляют собой точки (тривиальные акты комплексирования). В этом случае копределы по существу являются $c\text{-}DESC$-изоморфизмами (см. доказательство предложения 3), а φ — $r\text{-}DESC$-морфизмом. В силу условия (vii) определения 3 весь граф можно рассматривать как образ $r\text{-}DESC$-диаграммы относительно функтора $\ulcorner - \urcorner$, причем всегда существует единственный $r\text{-}DESC$-морфизм r, превращающий ее в коммутативную — он совпадает с φ с точностью до изоморфизма.

Более общий способ перевода предложен в [7]: если имеется функтор

$$r\text{-}sig : r\text{-}DESC \to SIG$$

с такой же функцией объектов, что и sig, то граф трансформации можно преобразовать в $\mathbf{D}SIG$-диаграмму, действуя на горизонтальные морфизмы функтором $\mathbf{D}sig$, а на вертикальные — функтором $\mathbf{D}r\text{-}sig$. Однако при этом удастся рассмотреть только трансформации интерфейсов. Чтобы остаться на уровне моделей программ, нужны условия, при выполнении которых граф трансформации становится (коммутативной) $\mathbf{D}c\text{-}DESC$-диаграммой. Мы рассмотрим два варианта таких условий, которым удовлетворяют дисциплины, формализующие ряд известных технологий программирования.

Первый вариант предполагает, что категория $r\text{-}DESC$ является подкатегорией в $c\text{-}DESC$. Это означает, что трансформация любой модели задает ее включение в результат трансформации в качестве компонента. Поэтому, чтобы произвести покомпонентную

трансформацию системы, порожденной некоторой конфигурацией, трансформации компонентов нужно включить в состав этой конфигурации. Более формально, для диаграммы $\Delta : X \to c\text{-}DESC$ и естественного $c\text{-}DESC$-преобразования вида $\varphi : |\Delta| \to \Sigma$ обозначим через $\mathbf{K}\varphi$ $c\text{-}DESC$-диаграмму $\mathbf{K}\llcorner\langle\varphi, 1_{|X|}\rangle\lrcorner$ — отрисовку $\mathbf{D}c\text{-}DESC$-стрелки, порожденной естественным преобразованием φ. Постоянный функтор $0 : \mathbf{1} \hookrightarrow \mathbf{2}$, рассматриваемый как $(\mathbf{Cat} \downarrow \mathbf{D}c\text{-}DESC)$-морфизм $0 : \ulcorner|\Delta|\urcorner \hookrightarrow \llcorner\langle\varphi, 1_{|X|}\rangle\lrcorner$, порождает каноническое вложение $\mathbf{K}ic(0) : |\Delta| \hookrightarrow \mathbf{K}\varphi$, а функтор $1 : \mathbf{1} \hookrightarrow \mathbf{2}$ — вложение $\mathbf{K}ic(1) : \Sigma \hookrightarrow \mathbf{K}\varphi$. $\mathbf{D}c\text{-}DESC$-диаграмма $\iota_\Delta : \Delta \hookleftarrow |\Delta| \hookrightarrow \mathbf{K}\varphi : \mathbf{K}ic(0)$, состоящая из канонических вложений, в силу предложения 2 имеет копредел, объект которого мы будем называть натяжкой (pull) диаграммы Δ семейством $c\text{-}DESC$-морфизмов φ и обозначать $\Delta \rightrightarrows \varphi$. Граф натяжки имеет вид «ежа», полученного из графа диаграммы Δ путем восставления из каждой вершины I стрелки-«иголки» $\varphi_I : \Delta(I) \to \Sigma(I)$. Используя канонические вложения $\Delta \hookrightarrow (\Delta \rightrightarrows \varphi) \hookleftarrow \mathbf{K}\varphi$, представляющие собой ребра копредела, можно определить следующий способ трансформаций конфигураций, гарантирующий коммутативность их графов:

$$\begin{array}{ccccc}
|\Delta| & \longrightarrow & \Delta & \xrightarrow{\operatorname{colim}\Delta} & \ulcorner colim(\Delta)\urcorner \\
{\scriptstyle\langle\varphi, 1_{|sch(\Delta)|}\rangle}\downarrow & & & & \downarrow{\scriptstyle\ulcorner colim(\Delta\hookrightarrow(\Delta\rightrightarrows\varphi))\urcorner} \\
\Sigma & \longrightarrow \boldsymbol{K}\varphi \longrightarrow & \Delta \rightrightarrows \varphi & \xrightarrow{\operatorname{colim}\Delta\rightrightarrows\varphi} & \ulcorner colim(\Delta \rightrightarrows \varphi)\urcorner
\end{array}$$

Определение 6. Пусть $\langle c\text{-}DESC, Conf, sig\rangle$ — формальная дисциплина специфицирования, $hr\text{-}DESC$ – подкатегория в $c\text{-}DESC$, содержащая все изоморфизмы. Если для любых диаграммы $\Delta \in Conf$ и естественного $hr\text{-}DESC$-преобразования вида $\varphi : |\Delta| \to \Sigma$ выполняются условия $\Delta \rightrightarrows \varphi \in Conf$ и $colim(\Delta \hookrightarrow (\Delta \rightrightarrows \varphi)) \in \operatorname{Mor} hr\text{-}DESC$, то четверка $\langle c\text{-}DESC, Conf, sig, hr\text{-}DESC\rangle$ называется (*трансформационно*) *однородной дисциплиной.*

Предложение 12. *Любая однородная дисциплина является формальной дисциплиной проектирования.*

Однородную дисциплину проектирования можно формально построить по заданной дисциплине специфицирования, используя конструкции, при помощи которых строятся факторизационные системы в категориях ([12], также [19]). Основной конструкцией является отношение Difip (diagonal fill-in property): напомним, что для морфизмов e, f выполняется $\operatorname{Difip}(e, f)$, если для всякого коммутативного квадрата $f \circ u = m \circ e$ существует единственный «диагональный» морфизм d такой, что $d \circ e = u$ и $f \circ d = m$. Для произвольного класса морфизмов M вводится обозначение $\mathrm{M}^{\uparrow\not=} = \{e \mid \operatorname{Difip}(e, \mathrm{M})\}$.

Предложение 13. *Пусть $\langle c\text{-}DESC, Conf, sig\rangle$ — произвольная формальная дисциплина специфицирования,* M *— произвольный класс $c\text{-}DESC$-морфизмов. Если класс $Conf$ замкнут относительно натяжек $\mathrm{M}^{\uparrow}$-морфизмами, то четверка*

$$\langle c\text{-}DESC, Conf, sig, (\operatorname{Ob} c\text{-}DESC, \mathrm{M}^{\uparrow})\rangle$$

является трансформационно однородной дисциплиной.

ДОКАЗАТЕЛЬСТВО. Условие $\operatorname{Difip}(\operatorname{Iso} c\text{-}DESC, \mathrm{M})$ выполняется для любого класса

M, поэтому $\mathrm{Iso}\, c\text{-}DESC \subseteq \mathrm{M}^{\uparrow}$. Остальные условия определения 6 проверяются непосредственно. □

Имеет место соотношение $\mathrm{Iso}\, c\text{-}DESC = (\mathrm{Mor}\, c\text{-}DESC)^{\uparrow}$, определяющее тривиальные трансформации, и $\mathrm{Mor}\, c\text{-}DESC = (\mathrm{Iso}\, c\text{-}DESC)^{\uparrow}$, определяющее возможность выбора $r\text{-}DESC = c\text{-}DESC$. Также можно показать, что если функтор sig имеет, в дополнение к левому, правый сопряженный с тождественной коединицей (т. е. любой интерфейс имеет «антидискретную» реализацию, поддерживающую все его возможности по включению в себя компонентов), то класс $sig^{-1}(\mathrm{Iso}\, SIG)$ имеет вид $\mathrm{M}^{\uparrow}/$для подходящего M (см. [12]); в частности, трансформациями могут служить в точности все перегрузки (обогащения структуры). Обычно однородные дисциплины возникают в контексте алгебраического подхода к проектированию систем, где и интеграция, и трансформация сводятся к обогащению вычислительных возможностей модели, описываемых множеством термов некоторой алгебры (типа данных). Примером однородной дисциплины с $r\text{-}DESC = c\text{-}DESC$ служит дисциплина **GAMMA**, рассмотренная в [7] (без явной формулировки свойства однородности). Модель программы в ней представляет собой разновидность машины переписывания термов (term rewriting), а действие по интеграции или трансформации — расширение множества правил переписывания с одновременным применением сигнатурного гомоморфизма типов данных.

Другой вид дисциплин, в которых трансформации конфигураций являются $\mathbf{D}c\text{-}DESC$-диаграммами, — двойственный к однородным: в них $r\text{-}DESC$ является подкатегорией в $c\text{-}DESC^{op}$ (напомним, что последняя получается из $c\text{-}DESC$ обращением направления всех морфизмов). Отметим, что такая ситуация не противоречит условию (vii) определения 3, поскольку двойственный к изоморфизму отождествляется с обратным к нему [12]. Здесь результаты трансформаций входят в исходные модели, так что появляется возможность трассирования — прослеживания трансформаций в обратном порядке в целях выявления назначения отдельных фрагментов моделей. Покомпонентная трансформация конфигурации системы может оставлять последнюю неизменной: результаты трансформаций уже содержатся в компонентах, а значит, и в системе. Действительно, для диаграммы $\Delta : X \to c\text{-}DESC$ и естественного $c\text{-}DESC$-преобразования вида $\tau : \Sigma \to |\Delta|$ (двойственность!) обозначим, как выше, через $\mathbf{K}\tau$ $c\text{-}DESC$-диаграмму $\mathbf{K}\llcorner\langle\tau, 1_{|X|}\rangle\lrcorner$. Имеются канонические вложения $\mathbf{K}ic(0) : \Sigma \hookrightarrow \mathbf{K}\tau, \mathbf{K}ic(1) : |\Delta| \hookrightarrow \mathbf{K}\tau$. Объект копредела $\mathbf{D}c\text{-}DESC$-диаграммы $\iota_\Delta : \Delta \hookleftarrow |\Delta| \hookrightarrow \mathbf{K}\tau : \mathbf{K}ic(1)$ будем называть *накачкой* (push) диаграммы Δ семейством τ и обозначать $\tau \rightrightarrows \Delta$. Граф накачки можно представлять как «мишень для стрельбы из лука» — он получается из графа диаграммы Δ путем «попадания» в каждую вершину «стрелой» из семейства τ. Пусть $\kappa : \Delta \hookrightarrow (\tau \rightrightarrows \Delta) \hookleftarrow \mathbf{K}\tau : \kappa'/$— ребра копредела. $(\mathbf{Cat} \downarrow \mathbf{D}c\text{-}DESC)$-морфизм $\langle\tau, !_{\mathbf{2}} : \mathbf{2} \to \mathbf{1}\rangle : \llcorner\langle\tau, 1_{|X|}\rangle\lrcorner \to \ulcorner|\Delta|\urcorner$ является левым обратным к морфизмам 0 и 1, откуда, в частности, $1_{|\Delta|}/\circ\, \iota_\Delta = (\iota_\Delta \circ \mathbf{K}ic(\langle\tau, !_{\mathbf{2}}\rangle)) \circ \mathbf{K}ic(1)$. По определению копредела получаем, что существует $\mathbf{D}c\text{-}DESC$-морфизм $\tau \rhd \Delta : (\tau \rightrightarrows \Delta) \to \Delta$ такой, что $(\tau \rhd \Delta) \circ \kappa = 1_{|\Delta|}/$и $(\tau \rhd \Delta) \circ \kappa'/ = \iota_\Delta \circ \mathbf{K}ic(\langle\tau, !_{\mathbf{2}}\rangle)$. Непосредственно проверяется, что $\mathrm{colim}\,(\tau \rightrightarrows \Delta) = \mathrm{colim}\,\Delta \circ (\tau \rhd \Delta)$, так что любая накачка диаграммы Δ имеет копредел с тем же объектом, что и Δ. Трансформации, двойственные к действиям по

интеграции, являются неразрушающими (non-invasive) по отношению к сборке систем: если $\Delta \in Conf$, τ состоит из двойственных к трансформациям и $(\tau \rightrightarrows \Delta) \in Conf$, то условие (viii) определения 3 можно обеспечить, взяв накачку в качестве $\Delta \oplus \tau^{op}$.

$$\begin{array}{ccccccc} |\Delta| & \longrightarrow & \Delta & & \xrightarrow{\operatorname{colim}\Delta} & \ulcorner colim(\Delta)\urcorner \\ {\scriptstyle\langle\tau,1_{|sch(\Delta)|}\rangle}\uparrow & & & & & \| \\ \Sigma & \longrightarrow \boldsymbol{K}\tau \longrightarrow & \tau \rightrightarrows \Delta & \xrightarrow{\tau\triangleright\Delta} & \Delta & \xrightarrow{\operatorname{colim}\Delta} & \ulcorner colim(\Delta)\urcorner \end{array}$$

Определение 7. Пусть $\langle c\text{-}DESC, Conf, sig\rangle$ — формальная дисциплина специфицирования, $hr\text{-}DESC$ — подкатегория в $c\text{-}DESC$, содержащая все изоморфизмы. Если класс $Conf$ замкнут относительно накачек $hr\text{-}DESC$-морфизмами, то четверка

$$\langle c\text{-}DESC, Conf, sig, hr\text{-}DESC^{op}\rangle$$

называется (*трансформационно*) *кооднородной* дисциплиной.

Предложение 14. *Любая кооднородная дисциплина является формальной дисциплиной проектирования.*

Замкнутыми относительно любых накачек являются: класс всех конечных диаграмм, всех коконусов, всех копроизведений коконусов (в частности, дисциплина моделирования сценариев SM кооднородна), а также класс всех допустимых конфигураций $\mathbf{D}sig^{-1}(ICMax_{sig})$ (см. предложение 6).

Кооднородность естественным образом возникает при построении дисциплин над **Set**. Действие трансформации $t^{op} : X \to Y$, двойственной по отношению к отображению $t : |Y| \to |X|$, можно описать как раскрытие (expansion) точек множества $|X|$ в подмножества множества $|Y|$, образующие его разбиение, с (частичным) переносом структуры объекта X на них. Точку множества $|X|$ можно рассматривать как обозначение класса задач, реализуемого путем раскрытия, в соответствие с интуитивным пониманием трансформации [15]. Например, в различных дисциплинах моделирования предметных областей посредством графов (таких как семантические сети, схемы рабочих процессов и т. д.) трансформации состоят в углублении предметных знаний, приводящем к замене вершин целыми подграфами.

6. Синтез дисциплин конфигурирования

Специализированные технологии комплексирования разрабатываются в целях повышения эффективности синтеза различных классов систем, аналогично тому, как создаются технологии программирования. При этом сложные технологии могут комплексироваться из простых так же, как программные системы. Для обоснования этого факта мы построим формальную дисциплину проектирования, объектами в которой служат формальные дисциплины конфигурирования (т. е. все пары вида $\langle c\text{-}DESC, Conf\rangle$, где $Conf$ — класс $c\text{-}DESC$-диаграмм, имеющих копределы). Ясно, что действием по интеграции $cm : \langle c\text{-}DESC_1, Conf_1\rangle \to \langle c\text{-}DESC_2, Conf_2\rangle$ может служить любой функтор $cm : c\text{-}DESC_1 \to c\text{-}DESC_2$, сохраняющий конфигурации (выполняется условие

$cm \circ Conf_1 \subseteq Conf_2$) и естественный относительно комплексирования систем (cm поднимает копределы всех диаграмм из $Conf_1$). Из предложения 4 вытекает, что такой функтор детерминирует, в частности сохраняет, копределы всех конфигураций. Совокупность всех формальных дисциплин конфигурирования и всех действий по их интеграции (со стандартным законом композиции функторов) образует категорию, которую мы обозначим через **CONF**.

Поскольку действия по интеграции дисциплин конфигурирования задаются на их категориях моделей, интерфейсом дисциплины $\langle c\text{-}DESC, Conf\rangle$ служит категория $c\text{-}DESC$. Непосредственно проверяется, что отображение $\langle c\text{-}DESC, Conf\rangle \mapsto c\text{-}DESC$ является функцией объектов функтора, «забывающего» конфигурации, который мы обозначим через $\boldsymbol{desc} : \mathbf{CONF} \to \mathbf{CAT}$. Возможности комплексирования дисциплин определяются классом $ICMax_{\boldsymbol{desc}}$, который имеет следующий состав.

Предложение 15. *Диаграмма $\Sigma : X \to \mathbf{CAT}$, имеющая копредел, входит в класс $ICMax_{\boldsymbol{desc}}$ тогда и только тогда, когда любое ребро ее копредела $\sigma_I, I \in \operatorname{Ob} X$, детерминирует копределы всех $\Sigma(I)$-диаграмм, детерминируемые всеми функторами $\Sigma(f)$, где f — X-морфизм с областью I.*

ДОКАЗАТЕЛЬСТВО. Рассмотрим диаграмму $\Sigma : X \to \mathbf{CAT}$ с копределом $\langle \sigma, !_X\rangle : \Sigma \to \ulcorner S \urcorner$. Если она удовлетворяет условию предложения, то любая **CONF**-диаграмма $\Xi \in \mathbf{D}\boldsymbol{desc}^{-1}(\{\Sigma\})$ имеет копредел $\langle \sigma, !_X\rangle : \Xi \to \ulcorner \langle S, \bigcup\!\!\bigwedge_{I \in \operatorname{Ob} X} \sigma_I \circ Conf_I\rangle \urcorner$, где $Conf_I$ — класс конфигураций дисциплины $\Xi(I)$. Допустим теперь, что имеются X-объект I и $\Sigma(I)$-диаграмма Δ такие, что все функторы $\Sigma(f)$, где $f \in \operatorname{Mor} X$ и $\operatorname{dom} f = I$, детерминируют копределы Δ, но σ_I не детерминирует их. Тогда диаграмма $\Theta : X \to \mathbf{CONF} : J \mapsto \langle \Sigma(J), \{\Sigma(f) \circ \Delta \mid f : I \to J \in \operatorname{Mor} X\}\rangle, h \mapsto \Sigma(h)$ не имеет копредела вида $\langle \sigma, !_X\rangle : \Sigma \to \ulcorner CF \urcorner$ с $\boldsymbol{desc}(CF) = S$, так что $\boldsymbol{desc}$ не поднимает копределы Θ, несмотря на то, что $\boldsymbol{desc} \circ \Theta = \Sigma$. □

В класс $ICMax_{\boldsymbol{desc}}$ входят, в частности, все дискретные **CAT**-диаграммы. Приведем пример **CAT**-диаграммы, не принадлежащей ему. Пусть C — частично упорядоченное множество $a \to c \leftarrow b$ (с несравнимыми элементами a и b), Σ — **CAT**-диаграмма вложений $C \hookleftarrow C\backslash\{c\} \hookrightarrow C$. Дискретная диаграмма $\{a, b\}$ имеет копредел в C, однако не имеет его в объекте копредела диаграммы Σ.

Трансформации дисциплин должны отражать реализацию систем, т. е. переход от дисциплин конфигурирования интерфейсов к дисциплинам конфигурирования их реализаций. В силу соображений, приведенных в начале разд. 3, некоторый функтор $sig : c\text{-}DESC \to SIG$ можно интерпретировать как правило выделения интерфейсов моделей дисциплины $\langle c\text{-}DESC, Conf\rangle$, если он удовлетворяет условиям (ii)–(v) определения 3. Его можно рассматривать как **CONF**-морфизм $sig : \langle c\text{-}DESC, Conf\rangle \to \langle SIG, sig \circ Conf\rangle$, удовлетворяющий условиям (ii), (iii), (v) (ср. пример, приведенный после предложения 7). Обозначим через $r\text{-}CONF$ класс всех таких **CONF**-морфизмов. Ясно, что он замкнут относительно композиций.

Определение 8. *Дисциплиной синтеза дисциплин конфигурирования* называется четверка

$$\boldsymbol{SCONF} = \langle \mathbf{CONF}, \mathbf{D}\boldsymbol{desc}^{-1}(ICMax_{\boldsymbol{desc}}), \boldsymbol{desc}, (\mathrm{Ob}\,\mathbf{CONF}, r\text{-}CONF)^{op} \rangle.$$

Предложение 16. $\boldsymbol{SCONF}$ *является кооднородной дисциплиной.*

ДОКАЗАТЕЛЬСТВО. Очевидно, что функтор $\boldsymbol{desc}$ унивалентен. Левый сопряженный к нему сопоставляет категории $c\text{-}DESC$ дисциплину конфигурирования $\langle c\text{-}DESC, \emptyset \rangle$, единица этого сопряжения тождественна. Далее применяем предложение 6. Осталось заметить, что в силу предложения 7 класс $r\text{-}CONF$ содержит все **CONF**-изоморфизмы. □

Аналогичным образом можно строить дисциплины синтеза дисциплин специфицирования и проектирования. В частности, совокупность всех дисциплин проектирования можно превратить в категорию, описывающую их интеграцию, если определить морфизмы дисциплин следующим образом.

Определение 9. *Морфизмом формальной дисциплины проектирования*

$$\langle c\text{-}DESC_1, Conf_1, sig_1 : c\text{-}DESC_1 \to SIG_1, r\text{-}DESC_1 \rangle$$

в дисциплину

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

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 поднимает копределы всех диаграмм из $Conf_1$;

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

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

Пара функторов $\langle cm, sm \rangle$, удовлетворяющая условиям (i)–(iii), называется *морфизмом формальной дисциплины 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 условиям (i)–(ii), называется *морфизмом формальной дисциплины конфигурирования* $\langle c\text{-}DESC_1, Conf_1 \rangle$ в $\langle c\text{-}DESC_2, Conf_2 \rangle$.

При помощи этих конструкций вводятся категории дисциплин **ARCH** и **SPEC** по аналогии с **CONF**. Имеется цепочка забывающих функторов

$$\mathbf{CAT} \leftarrow \mathbf{CONF} \leftarrow \mathbf{SPEC} \leftarrow \mathbf{ARCH}.$$

Детальное рассмотрение соответствующих дисциплин проектирования выходит за рамки настоящей работы. Отметим, что построение дисциплины синтеза дисциплин специфицирования можно описать как формальное преобразование дисциплины $\boldsymbol{SCONF}$,

воспроизводящее переход от модульного проектирования к аспектно-ориентированному, описанное в [17]. Это можно истолковать так, что категория интерфейсов отражает аспектную структуру реализаций (например, ассортимент ролей, которые компоненты играют в составе систем [20]). Аспектами здесь являются все дисциплины, категория интерфейсов у которых сингулярна (их характеризация дана в предложении 8). Забывающий функтор $\boldsymbol{conf} : \mathbf{SPEC} \to \mathbf{CONF}$ служит правилом выделения интерфейсов у дисциплин специфицирования. В частности, он является унивалентным и имеет левый сопряженный, сопоставляющий дисциплине конфигурирования $\langle c\text{-}DESC, Conf\rangle$ дисциплину специфицирования $\langle c\text{-}DESC, Conf, 1_{c\text{-}DESC}\rangle$ (предложение 7). Забывающий функтор $\boldsymbol{spec} : \mathbf{ARCH} \to \mathbf{SPEC}$, напротив, не может служить правилом выделения интерфейсов, поскольку он не унивалентен.

Заключение

С точки зрения научной прагматики, теория категорий почти не производит нового знания. Она предназначена для выявления общности внешне различных математических конструкций без потери строгости и проведения однократных доказательств утверждений о них. Результаты такого рода в области моделирования синтеза программных систем, в том числе представленные в настоящей работе, способны упростить применение мультипарадигменных подходов наподобие MDA и свести к минимуму вынужденное уменьшение числа используемых технологий программирования, вызванное трудностями интеграции их артефактов. Этого добиваются автор и его коллеги при создании прикладных крупномасштабных систем [16].

Важным источником конструкций, теоретико-категорное представление которых формально описывает синтез систем, должна служить мереология, анализизирующая отношения «часть-целое» [21]. В частности, копредел отвечает мереологической сумме. В состоятельной формальной мереологии основные принципы комплексирования систем должны иметь вид аксиом и теорем. Претендентом на роль мереологической теоремы может выступать наблюдение из разд. 1, что любой акт комплексирования комбинируется из трех приемов интеграции части в целое: изменение части (загрузка), изменение целого (подстановка) и добавление «клея» (соединение). Ему можно сопоставить следующий теоретико-категорный факт: если категория имеет инициальный объект (универсальную единицу загрузки) и в ней любая пара стрелок с общим началом (диаграмма соединения) имеет кодекартов квадрат (ребра которого образуют диаграмму подстановки), то любая конечная диаграмма имеет в ней копредел (см. двойственное утверждение в [12. Теорема 12.4(3)]). Так что в целом настоящую работу можно рассматривать как опыт введения в «алгебраическую мереологию».

Список литературы

1. *Дал У., Дейкстра Э., Хоор К.* Структурное программирование. М.: Мир, 1975.
2. *Буч Г.* Объектно-ориентированный анализ и проектирование с примерами прило-

жений на C++. 2-е изд. М.: Бином; СПб.: Невский диалект, 1999.

3. *Kiczales G. et al.* Aspect-Oriented Programming // Lecture Notes in Computer Sci. 1997. Vol. 1241. P. 220–242.

4. *Frankel D. S.* Model Driven Architecture: Applying MDA to Enterprise Computing. N. Y.: Wiley and Sons, 2003.

5. *Маклейн С.* Категории для работающего математика. М.: Физматлит, 2004.

6. *Соммервилл И.* Инженерия программного обеспечения. 6-е изд. М.: Вильямс, 2002.

7. *Fiadeiro J. L., Lopes A., Wermelinger M.* A Mathematical Semantics for Architectural Connectors // Lecture Notes in Computer Sci. 2003. Vol. 2793. P. 190–234.

8. *Pinto M., Fuentes L., Troya J. M.* DAOP-ADL: An Architecture Description Language for Dynamic Component and Aspect-Based Development // Lecture Notes in Computer Sci. 2003. Vol. 2830. P. 118–137.

9. *Гамма Э., Хелм Р., Джонсон Р., Влассидес Дж.* Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб.: Питер, 2001.

10. *Lopes A., Fiadeiro J. L.* Revisiting the Categorical Approach to Systems // Lecture Notes in Computer Sci. 2002. Vol. 2422. P. 426–440.

11. *Allen R. J., Garlan D.* A Formal Basis for Architectural Connection // ACM Trans. on Software Engineering and Methodology. 1997. Vol. 6 (3). P. 213–249.

12. *Adámek J., Herrlich H., Strecker G.* Abstract and Concrete Categories. N. Y.: Wiley and Sons, 1990.

13. *Bednarczyk M. A., Borzyszkowski A. M., Pawlowski W.* Epimorphic Functors. Gdansk: Institute of Computer Science, 2007. URL: http://www.ipipan.gda.pl/andrzej/papers/categ-ipi.ps.gz.

14. *Guitart R., van den Bril L.* Décompositions et Lax-complétions // Cahiers de Topologie et Géométrie Différentielle Catégoriques. 1977. T. 18. No. 4. P. 333–407.

15. *Sannella D.* A Survey of Formal Software Development Methods // Software Engineering: A European Prospective. IEEE Computer Society Press, 1993. P. 281–297.

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

17. *Ковалёв С. П.* Формальный подход к аспектно-ориентированному моделированию сценариев // Сиб. журн. индустр. математики. 2010. Т. 13, № 3. С. 30–42.

18. *Соренсен И.* Язык спецификаций // Требования и спецификации в разработке программ. М.: Мир, 1984. С. 223–239.

19. *Carboni A., Janelidze G., Kelly G. M., Paré R.* On Localization and Stabilization for Factorization Systems // Applied Categorical Structures. 1997. Vol. 5. P. 1–58.

20. *Hanenberg S., Unland R.* Roles and Aspects: Similarities, Differences, and Synergetic Potential // Lecture Notes in Computer Sci. 2002. Vol. 2425. P. 507–520.

21. *Le D. T. M., Janicki R.* A Categorical Approach to Mereology and its Application to Modelling Software Components // Lecture Notes in Computer Sci. 2008. Vol. 5084. P. 146–174.

Материал поступил в редколлегию 17.06.2011

Адрес автора
КОВАЛЁВ Сергей Протасович
Институт проблем управления им. В. А. Трапезникова РАН
ул. Профсоюзная, 65, Москва, 117997, Россия
e-mail: kovalyov@nm.ru