ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ № 3(17), 2013, с.1—33

УДК 519.87

Ю.И. Бродский

Синтез многокомпонентных имитационных моделей и модельно-ориентированное программирование[1]

Аннотация. Работа посвящена проблеме описания и моделирования сложных систем, таких, про которые хорошо известно из каких компонент они состоят, что эти компоненты умеют делать, по каким правилам взаимодействуют между собой. Проблемой моделирования, притом весьма непростой, является воспроизведение поведения и оценка возможностей сложной системы в целом.

Предлагается новый подход к проектированию и компьютерной реализации имитационных моделей сложных многокомпонентных систем, отличающийся от господствующего в последние десятилетия объектного подхода. Он назван в работе модельным анализом и модельно-ориентированным программированием. Центральным понятием этого подхода и в то же время элементарным кирпичиком для построения любых более сложных конструкций является понятие модели-компоненты. Модель-компонента наделена более сложной структурой, чем, например, объект объектного анализа. Эта структура обеспечивает модели-компоненте поведение – способность стандартным образом отвечать на стандартные запросы ее внутренней и внешней среды.

Организация имитационных вычислений модели-компоненты оказывается инвариантной относительно операции объединения моделей-компонент в модель-комплекс, поскольку модель-комплекс сохраняет структуру модели-компоненты. Это позволяет, во-первых, строить фрактальные модели любой сложности и, во-вторых, реализовывать вычислительный процесс даже очень сложных моделей единообразно - единой программой. Кроме того, предлагаемый подход к компьютерной реализации многокомпонентных имитационных моделей полностью исключает императивное программирование, и позволяет производить программный код высокой степени параллельности.

Ключевые слова и фразы: Имитационное моделирование, многокомпонентные сложные системы, модельно-ориентированное программирование, параллельные и распределенные вычисления

---

[1] Работа выполнена при финансовой поддержке РФФИ, грант № 13-01-00499-а и РГНФ, грант 12-06-00932-а.

Введение

Во введении на неформальном, «гуманитарном» уровне дается обзор проблематики, основных идей, методов и результатов данной работы, дабы в дальнейшем они не были заслонены формальными построениями.

В плане неформального понимания того, что такое сложная многокомпонентная система, автору очень близки следующие три высказывания видного специалиста в области сложных систем и их имитационного моделирования, член.-корр. АН СССР Николая Пантелеймоновича Бусленко, почерпнутые из работ [1], [2]: «Сложная система – составной объект, части которого можно рассматривать как системы, закономерно объединённые в единое целое в соответствии с определенными принципами или связанные между собой заданными отношениями», «В каждый момент времени элемент сложной системы находится в одном из возможных состояний; из одного состояния в другое он переходит под действием внешних и внутренних факторов», «Для построения синтеза поведения сложной системы необходимо дать ее компонентам возможность в полной мере проявить себя». В некотором смысле все, что предложено в данной работе, является одной из возможных реализаций приведенных выше высказываний.

Полностью соглашаясь на интуитивном уровне с тем, что сложная система может сама состоять из сложных систем, мы не беремся давать здесь строгое определение этому понятию. Тем не менее, понятие модели сложной системы будет введено далее вполне формально, как род структуры в смысле Н. Бурбаки [3]. С этой точки зрения сложная система – это то, что достаточно адекватно может быть представлено моделью сложной системы.

Род структуры – развитие понятия множества. Базисное множество снабжается структурой некоторого рода – вводится определенный тип отношений между его элементами, и в зависимости от этого типа отношений, множество может стать, например, группой, или решеткой, или векторным пространством, или же в нашем случае – имитационной моделью сложной системы. При этом математический объект, например конкретное линейное пространство, является экземпляром структуры соответствующего рода.

Метод структурализма, идейно восходящий к Эрлангенской программе Ф. Клейна, и оказавшийся достаточно популярным и продуктивным в XX веке, в том числе и в гуманитарных науках, предлагает далее рассматривать различные преобразования базисных множеств и искать инварианты этих преобразований.

Школа член-корр. РАН Ю.Н. Павловского в ВЦ РАН в последние десятилетия развивала геометрическую теорию декомпозиции [4]-[7], где с помощью морфизмов пытаются найти более простые представления различных математических объектов – их редукции и декомпозиции.

В данной работе, посвященной синтезу модели сложной системы из моделей ее компонент, нас более всего будет интересовать возможность распространения рода структуры на объединение базисных множеств математических объектов, наделенных этим родом структуры. Инвариантом, относительно объединения базисных множеств имитационных моделей, оказывается предложенный ниже способ организации имитационных вычислений.

Модельный синтез и анализ, как способ описания и синтеза имитационных моделей сложных многокомпонентных систем, развивался в отделе Имитационных систем ВЦ РАН с конца 80-х гг. Основные его идеи и методы изложены в работах [8]-[12], однако сам термин «модельный синтез» впервые был предложен в работе [8]. В основе модельного синтеза и анализа лежит понятие модели-компоненты.

Модель-компонента – это, прежде всего, имитационная модель. Как имитационная модель она имеет внутренние и внешние характеристики и удовлетворяет гипотезе о замкнутости и предположению о наблюдаемости внешних характеристик.

Гипотеза о замкнутости модели постулирует, что знания значений внутренних и внешних характеристик модели в момент t достаточно для вычисления ее внутренних характеристик на некотором интервале $(t, t+\Delta t)$. Таким образом, внутренние характеристики с одной стороны определяют состояние модели, а с другой стороны могут прогнозироваться с помощью этой модели на некотором интервале времени. Внешние характеристики не моделируются, но влияют на модель. Поэтому их можно считать доступными для восприятия модели характеристиками внешнего по отношению к ней мира (в том числе, например, внешними управляющими воздействиями). Обычно предполагается, что внешние характеристики доступны для измерения в любой момент модельного времени. Иногда предположение о наблюдаемости внешних характеристик включают в гипотезу о замкнутости. В данной работе, упоминая гипотезу о замкнутости, всегда будем считать, что предположение о наблюдаемости внешних характеристик включается в нее.

С точки зрения построения программных систем, модель-компонента подобна объекту объектного анализа, но помимо характеристик, снабженному не только методами, способными делать что-то полезное, если их вызовут, а неким аналогом операционной системы, всегда готовым давать стандартные ответы на стандартные запросы внутренней и внешней среды модели.

С формальной точки зрения, модель-компонента есть математический объект, базисным множеством которого является совокупность множеств внутренних и внешних характеристик модели, методов (того, что модель умеет делать) и событий (того, на что модель должна уметь реагировать). На базисном множестве вводится род структуры «модель-компонента», который обладает двумя замечательными свойствами:

1. Род структуры «модель-компонента» позволяет стандартным и однозначным образом организовать вычислительный процесс моделирования для всех объектов, снабженных структурой этого рода. Это означает возможность создания универсальной программы, способной запустить на выполнение любую имитационную модель, если та является математическим объектом, снабженным структурой рода «модель-компонента».

2. Вообще говоря, если рассмотреть два произвольных математических объекта снабженных структурой одного рода (например, структурой абстрактной группы), то распространение этой структуры на объединение их базисных множеств возможно далеко не всегда. Тем не менее, для рода структуры «модель-компонента», подобное распространение общей структуры компонент на объединение их базисных множеств или возможно (если подмножества характеристик их базисных множеств не имеют попарных пересечений), или возможно с некоторыми оговорками (например, при условии пополнения исходных объектов-компонент некоторым количеством дополнительных объектов-компонент, снабженных той же структурой).

Второе свойство позволяет образовывать из моделей-компонент путем объединения их базисных множеств модели-комплексы, которые после распространения общей структуры компонент на объединение их базисных множеств оказываются математическими объектами того же самого рода структуры «модель-компонента», и стало быть, снова могут объединяться в модели-комплексы. Первое свойство позволяет не впадать в отчаяние от сложности вычислительного процесса, получающейся в результате таких объединений сверхсложной фрактальной модели.

Для программной реализации сложной многокомпонентной модели предлагается модельно-ориентированная парадигма программирования, где единицей проектирования программного комплекса является модель-компонента – конструкция более агрегированная по сравнению с объектом объектного анализа.

Ниже будет показано, что модельно-ориентированное программирование позволяет полностью исключить наиболее сложное как для разработки, так и для отладки, императивное программирование [12]. Кроме того, получаемый исполняемый код отличается высокой степенью параллельности, при этом степень параллельности кода возрастает при росте сложности модели. Данный факт может открыть перспективы для применения методов модельно-ориентированного программирования на высокопроизводительных вычислительных системах, в том числе и для задач не связанных с имитационным моделированием, но имеющих многокомпонентную организацию.

1. Роды структур

Оригинальное изложение аппарата родов структур, содержащееся в работе [3], опирается на специфическую «Бурбаковскую» терминологию и аксиоматику, отличающуюся от принятой в большинстве учебников по теории множеств. Тем не менее, существуют работы, например [5] и [13], где этот аппарат излагается на основании классической терминологии и аксиоматики. В русле именно этих работ мы и дадим здесь определение рода структуры.

Род структуры объявляет о том, что математический объект будет состоять из частей $\sigma_1,\ldots,\sigma_r$ некоторых множеств. Это записывается следующим образом

$$\sigma_1 \subset S_1(X_1,\ldots,X_n,A_1,\ldots,A_m),\ldots,\ \sigma_r \subset S_r(X_1,\ldots,X_n,A_1,\ldots,A_m).$$

Здесь множества $X_1,\ldots,X_n$ называются «базисными», множества $A_1,\ldots,A_m$ – вспомогательными (употребляемая терминология почерпнута из [3]). Множества $S_k(X_1,\ldots,X_n,A_1,\ldots,A_m)$, $1 \le k \le r$ – так называемые ступени, построенные по схеме S_k из исходных множеств $X_1,\ldots,X_n,A_1,\ldots,A_m$. Ступени получаются путем применения к исходным множествам и/или уже имеющимся ступеням операций декартова произведения $\times$, и взятия множества всех подмножеств $\beta(\cdot)$. А именно:

1. По определению X_i – ступень, при любом $1 \le i \le n$. Аналогично A_j – ступень, при любом $1 \le j \le m$.
2. Если S – ступень, то и $\beta(S)$ – ступень.
3. Если S и S' – ступени, то и $S \times S'$ – ступень.
4. Других ступеней нет.

Схема S_k фиксирует исходные множества и порядок применения к ним двух указанных выше операций.

Базисные и вспомогательные множества играют разную роль в построениях родов структур (например, в построениях данной работы вспомогательные множества вообще не используются). Базисные множества должны быть обозначены разными буквами. На обозначения вспомогательных множеств таких ограничений не накладывается. Соотношения

$$\sigma_1 \subset S_1(X_1,\ldots,X_n,A_1,\ldots,A_m),\ldots,\sigma_r \subset S_r(X_1,\ldots,X_n,A_1,\ldots,A_m)$$

называются соотношениями типизации. В род структур, кроме соотношений типизации может входить еще некоторое соотношение

$$R(X_1,\ldots,X_n,\sigma_1,\ldots,\sigma_r,A_1,\ldots,A_m,\xi_1,\ldots,\xi_s).$$

Это соотношение предъявляет к роду структуры некоторые требования. Здесь $\xi_1,\ldots,\xi_s$ – соотношения, касающиеся множеств $A_1,\ldots,A_m$. Вспомогательные множества $A_1,\ldots,A_m$ и соотношения $\xi_1,\ldots,\xi_s$ не преобразуются – отображения рассматриваются только над базисными множествами.

Соотношение $R(X_1,\ldots,X_n,\sigma_1,\ldots,\sigma_r,A_1,\ldots,A_m,\xi_1,\ldots,\xi_s)$ называется «аксиомой» данного рода структуры. К нему предъявляется требование: оно должно быть переносимо при биекциях. Это означает, что

$$(f_i; X_i \to X_i', i=1,2,\ldots,n,) \text{– биекции}) \Rightarrow$$

$$(R(X,\sigma,A,\xi) \Rightarrow R(X'\sigma',A,\xi)), \text{ где } \sigma' = S(f,id_A).$$

Здесь $S(f,id_A)$ – так называемое «распространение» [3], [4] отображений f базисных множеств и тождественное распространение вспомогательных множеств по схеме S.

Род структуры будет обозначаться далее следующим образом:

$$\Sigma(A_1,\ldots,A_n,\xi_1,\ldots,\xi_s) =$$
$$=< X_1,\ldots,X_n; [\sigma_1 \subset S_1(X,A),\ldots,\sigma_r \subset S_r(X,A)]; [R(X,\sigma,A,\xi)] >$$

или более коротко:

$$\Sigma[(A,\xi)] = < X; [\sigma \subset S(X,A)]; [R(X,\sigma,A,\xi)] >.$$

Ломаные скобки «<» и «>» здесь играют роль ограничителя. Необязательные элементы рода структуры, как это принято при записи такого сорта конструкций, поставлены в квадратные скобки. Далее по возможности будет использоваться короткая форма всех соотношений.

Пусть имеются множества (E,τ) и выполняются соотношения $\tau \subset S(E,A)$ и $R(E,\tau,A,\xi)$, т. е. соотношение типизации и аксиома рода структуры $\Sigma[(A,\xi)]$ для множеств (E,τ). Тогда говорят, что объект (E,τ) снабжен структурой τ рода $\Sigma[(A,\xi)]$ или, что τ есть структура рода $\Sigma[(A,\xi)]$ на множестве E. Обозначение $\Sigma[(A,\xi)]$ содержит то, что не меняется при переходе от одного $\Sigma[(A,\xi)]$-объекта к другому.

Это *означает*, что множества $A_1,\ldots,A_m$ и соотношения $\xi_1,\ldots,\xi_s$ являются одними и теми же для всех объектов данного рода структуры.

Тем самым, как уже говорилось, математические объекты делятся на классы. К классу относятся объекты, снабженные структурой данного рода $\Sigma(A,\xi)$.

2. Семейство родов структур «модель-компонента»

Попробуем формально определить математический объект, представляющий элементарную имитационную модель сложной системы. Предыдущий подраздел на неформальном уровне поясняет, почему ее стоит определять именно таким образом.

Введем однопараметрическое семейство родов структур Σ_N «модель-компонента». Параметром N семейства Σ_N является количество процессов модели-компоненты. Формально процессы модели-компоненты будет определены ниже соотношениями типизации (10) и аксиомами R_9.

$$\Sigma_N =< X, M, E, \{M_j\}_{j=1}^N, \{E_j\}_{j=1}^N;$$

$$x \subset X\ ,\ a \subset X\ ,\ s \subset M\ ,\ f \subset M\ ,$$

$$\{m_{j,real} \subset M_j \times M\}_{j=1}^N,$$

$$\{e_{j,real} \subset E_j \times E\}_{j=1}^N,$$

$$\{m_{j,in} \subset M_j \times \beta(X)\}_{j=1}^N,$$

$$\{m_{j,out} \subset M_j \times \beta(X)\}_{j=1}^N,$$

$$\{e_{j,in} \subset E_j \times \beta(X)\}_{j=1}^N,$$

$$\{m_j^0 \subset M_j\}_{j=1}^N,$$

$$\{sw_j \subset E_j \times M_j \times M_j\}_{j=1}^N,$$

$$\{p_j \subset \beta(M_j) \times \beta(E_j) \times M_j \times \beta(E_j \times M_j \times M)\}_{j=1}^N;$$

$$R_1:\ (x \cup a = X)\,\&\,(x \cap a = \varnothing),$$

$$R_2:\ (s \cup f = M)\,\&\,(s \cap f = \varnothing),$$

$$R_3:\ \{(\forall m \in M_j)(\exists!\tilde{m} \in M)(\{m, \tilde{m}\} \in m_{j,real})\}_{j=1}^N,$$

$$R_4:\ \{(\forall e \in E_j)(\exists!\tilde{e} \in E)(\{e, \tilde{e}\} \in e_{j,real})\}_{j=1}^N,$$

$$R_5:\ \{(\forall m \in M_j)(\exists!r \in \beta(X))(\{m, r\} \in m_{j,in})\}_{j=1}^N,$$

$$R_6:\ \{(\forall m \in M_j)(\exists!r \in \beta(x))(\{m, r\} \in m_{j,out})\}_{j=1}^N,$$

$$R_7:\ \{(\forall e \in E_j)(\exists!r \in \beta(X))(\{e, r\} \in e_{j,in})\}_{j=1}^N,$$

$$R_8: \left\{\left(\left(\forall e \in E_j\right)\left(\exists ! r \in M_j \times M_j\right)\left(\{e, r\} \in sw_j\right)\right)\& \right.$$
$$\left. \& \left(\left(\{e, r\} \in sw_j, \{\tilde{e}, \tilde{r}\} \in sw_j, r = \tilde{r}\right) \Rightarrow \left(e = \tilde{e}\right)\right)\right\}_{j=1}^{N}$$

$$R_9: \left\{p_j = \left\{ M_j, E_j, m_j^0, sw_j \right\}\right\}_{j=1}^{N},$$

R_{10}: аксиома однозначности вычисления характеристик модели-компоненты,

R_{11}: аксиома поведения модели-компоненты (организации имитационных вычислений)>.

Обозначение $\left\{\ldots_j\right\}_{j=1}^{N}$ используется для краткости и означает, что содержимое скобок повторяется через запятую N раз, при этом индекс j заменяется на $1,\ldots,N$. Например, $\left\{M_j\right\}_{j=1}^{N}$ есть краткий вариант записи $M_1,\ldots,M_N$.

Дадим пояснения приведенному выше формальному определению рода структуры «модель-компонента». Оно заключено в угловые скобки «<» и «>», и состоит из трех частей, которые отделяются друг от друга символом «;», а отдельные определения в каждой части разделяются символом «,». В первой части определяются базисные множества. Во второй части перечислены соотношения типизации, которые определяют родовые константы, как части множеств, построенных из базисных специальным образом – возможным последовательным применением операций декартова произведения $\times$ и взятия множества всех подмножеств $\beta(\cdot)$. В третьей части перечислены аксиомы рода структуры R_1 – R_{11}, которые устанавливают определенные зависимости между базисными множествами и родовыми константами.

В качестве основных базисных множеств выбраны:

$$X, M, E, \left\{M_j\right\}_{j=1}^{N}, \left\{E_j\right\}_{j=1}^{N}$$

Здесь X – множество характеристик модели, причем, иногда будем разделять его на два подмножества: $X=\{x,a\}$, где x — внутренние характеристики модели, то, что в соответствии с гипотезой о замкнутости полностью определяет ее состояние, а a — ее внешние характеристики, то, что в силу той же гипотезы о замкнутости полностью определяет взаимодействие модели с внешним по отношению к ней миром. Первое из соотношений типизации

$$x \subset X\ ,\ a \subset X\ , \tag{1}$$

утверждает, что внутренние характеристики модели x и внешние характеристики a — есть части ее характеристик. Аксиома R_1: $(x \cup a = X)\,\&\,(x \cap a = \varnothing)$ утверждает, что эти части x и a не пересекаются, и в совокупности составляют все множество характеристик модели X.

Далее идет M — множество различных реализаций методов-элементов, элементарных умений нашей модели, среди которых мы также иногда будем выделять два подмножества:

$$s \subset M\ ,\ f \subset M\ , \tag{2}$$

связанных аксиомой: R_2: $(s \cup f = M)\,\&\,(s \cap f = \varnothing)$. $M=\{s,f\}$ — медленные s, реализующие гладкую зависимость внутренних характеристик модели от ее внутренних и внешних характеристик, если например, $\dot{x}=F(x,a)$, то $x(t+\Delta t)\approx x(t)+F(x(t),a(t))\Delta t$, и быстрые f — реализующие скачки внутренних характеристик модели: $\Delta x = G(x(t),a(t))$.

E — множество различных реализаций методов-событий, связанных с моделью. События – это то, на что обязана реагировать наша модель. Метод-событие – функция, которая получая на входе подмножество характеристик $Y \subset X$, на выходе дает неотрицательное число, означающее, что событие наступило, если число равно нулю, или же прогноз времени до наступления события, если число положительно.

Отметим, что во множествах X, M, E нет одинаковых элементов. Этот факт также утверждается аксиомой рода структуры R_8. Для X, это следует из того, что речь идет о модели, и принцип неувеличения числа сущностей сверх необходимости (бритва Оккама) побуждает нас избегать ненужного дублирования одних и тех же характеристик, как моделируемого явления, так и его связей с внешним миром. Что касается M и E, словами «множество различных реализаций» подчеркивается именно уникальность методов в этих множествах. Множества M и E являются «хранилищами» различных функциональных зависимостей для модели-компоненты.

Далее будут упоминаться процессы модели-компоненты. Подробное объяснение того, что такое процессы будет дано далее, пока же скажем, что содержательно процессы подобны, например, системным службам операционной системы компьютера. Они работают всегда и при этом одновременно. Будем считать, что число процессов модели – N. Каждый процесс p_j, $j = 1,\ldots,N$ последовательно осуществляет некоторый конечный набор возможных для него элементарных действий M_j, который будем называть множеством его методов-элементов, возможно, в зависимости от возникающих в системе ситуаций E_j, на которые процесс умеет реагировать, их будем называть множеством его методов-событий.

Этим исчерпывается описание базисных множеств.

Вспомогательных множеств нет.

Продолжим рассмотрение соотношений типизации и аксиом.

$$\left\{ m_{j,real} \subset M_j \times M \right\}_{j=1}^{N} \qquad (3)$$

Для каждого процесса p_j задается отображение множества его методов-элементов M_j во множество их реализаций M. Таким образом, для любого $m \in M_j$ однозначно определяется его реализация $\tilde{m} \in M$. Формально это можно записать как аксиомы

$$R_3: \left\{(\forall m \in M_j)(\exists ! \tilde{m} \in M)(\{m, \tilde{m}\} \in m_{j,real})\right\}_{j=1}^N .$$

Разница между имеющими одинаковую реализацию методами, с точки зрения процесса, может быть, например, в способе коммутации их параметров с характеристиками модели, о чем речь пойдет ниже. Не запрещаются (хотя и не рекомендуются) и просто «синонимы».

$$\left\{e_{j,real} \subset E_j \times E\right\}_{j=1}^N \tag{4}$$

Для каждого процесса p_j задается отображение множества его методов-событий E_j во множество их реализаций E . Для любого $e \in E_j$ однозначно определяется его реализация $\tilde{e} \in E$. Аксиомы

$$R_4: \left\{(\forall e \in E_j)(\exists ! \tilde{e} \in E)(\{e, \tilde{e}\} \in e_{j,real})\right\}_{j=1}^N$$

утверждают этот факт. Так же как и в случае методов-элементов, различные с точки зрения процесса события, могут иметь одинаковые реализации.

$$\left\{m_{j,in} \subset M_j \times \beta(X)\right\}_{j=1}^N \tag{5}$$

Коммутируются внутренние и внешние характеристики модели с входящими параметрами методов-элементов j -го процесса. Включение должно определить, какое подмножество характеристик модели передается каждому из методов-элементов. Это можно выразить аксиомами

$$R_5: \left\{(\forall m \in M_j)(\exists ! r \in \beta(X))(\{m, r\} \in m_{j,in})\right\}_{j=1}^N .$$

$$\left\{m_{j,out} \subset M_j \times \beta(X)\right\}_{j=1}^{N} \qquad (6)$$

Коммутируются возвращаемые параметры методов-элементов с внутренними характеристиками модели. При этом должны выполняться аксиомы

$$R_6: \left\{(\forall m \in M_j)(\exists\, !r \in \beta(x))(\{m, r\} \in m_{j,out})\right\}_{j=1}^{N}.$$

$$\left\{e_{j,in} \subset E_j \times \beta(X)\right\}_{j=1}^{N} \qquad (7)$$

Коммутируются внутренние и внешние характеристики модели с входящими параметрами методов-событий j-го процесса. Включение должно определить, какое подмножество характеристик модели передается каждому из методов-событий. При этом должны выполняться аксиомы

$$R_7: \left\{(\forall e \in E_j)(\exists\, !r \in \beta(X))(\{e, r\} \in e_{j,in})\right\}_{j=1}^{N}.$$

Соотношение

$$\left\{m_j^0 \subset M_j\right\}_{j=1}^{N} \qquad (8)$$

определяет начальные методы-элементы процессов.

$$\left\{sw_j \subset E_j \times M_j \times M_j\right\}_{j=1}^{N} \qquad (9)$$

Определяются переключения методов-элементов в каждом процессе. Если при определенных условиях возможно переключение процесса с выполнения метода $A \in M_j$, на выполнение метода $B \in M_j$ (а таким условием, в силу гипотезы о замкнутости, может быть лишь некое сочетание внутренних и внешних характеристик модели $\{x, a\} \subset X$, – в виртуальном мире модели просто ничего больше нет), то должен быть создан единственный метод-событие

$e_{AB}(x,a) \in E_j$, прогнозирующий и вычисляющий такое переключение. Возможны также события вида $e_{AA}(x,a)$, не переключающие метод-элемент $A \in M_j$ на другой, а лишь прерывающие его выполнение (например, для синхронизации результатов его вычислений с другими процессами модели). Таким образом, время и порядок переключений элементов каждого процесса определяется тем, что происходит внутри и вне модели. Вообще говоря, в самых простых процессах, событий может и не быть, тогда не будет и переключений. Если же множество событий j-го процесса – E_j, непусто, то соотношение типизации (9) должно удовлетворять следующим аксиомам

$$R_8: \ \{((\forall e \in E_j)(\exists ! r \in M_j \times M_j)(\{e,r\} \in sw_j))\&$$
$$\&((\{e,r\} \in sw_j, \{\tilde{e},\tilde{r}\} \in sw_j, r = \tilde{r}) \Rightarrow (e = \tilde{e}))\}_{j=1}^{N},$$

утверждающим, что отображения множеств методов-событий во множества переключений инъективны.

$$\{p_j \subset \beta(M_j) \times \beta(E_j) \times M_j \times \beta(E_j \times M_j \times M)\}_{j=1}^{N} \qquad (10)$$

Соотношения типизации (10) определяют процессы модели, а соответствующий им набор аксиом

$$R_9: \ \{p_j = \{M_j, E_j, m_j^0, sw_j\}\}_{j=1}^{N},$$

уточняет это понятие. Процесс – это множества элементов и событий, начальный элемент и правила переключений. Множества M_j и E_j – это множества элементов и событий j-го процесса; m_j^0 – его начальный элемент; sw_j – правила переключения элементов, которые были подробно описаны выше.

Формальное определение семейства родов структур «модель-компонента» на этом заканчивается. Тем не менее, дополним набор аксиом семейства родов структур «модель-компонента» еще двумя.

Эти аксиомы в некотором смысле стоят особняком, поскольку ничего не добавляют к формальному определению отношений, задаваемых на базисных множествах рассматриваемого семейства родов структур – упомянутые отношения уже полностью определены. Смысл дополнительных аксиом $\boldsymbol{R}_{10}$ и $\boldsymbol{R}_{11}$ в том, чтобы описать, что и как мы собираемся делать дальше с математическими объектами семейства родов структур «модель-компонента».

Аксиома $\boldsymbol{R}_{10}$ требует однозначности вычисления характеристик модели-компоненты. Возможность однозначности вычисления характеристик постулируется гипотезой о замкнутости. Для модели-компоненты это требование означает, например, что два метода-элемента разных процессов не должны одновременно менять одну и ту же характеристику. А гипотеза о замкнутости утверждает, что этого возможно добиться. Собственно для модели-компоненты этим все и ограничивается, и можно было бы не выделять это требование в аксиому. Однако, при объединении компонент в комплекс, все оказывается не так просто, и эта аксиома начинает работать. Поэтому пока можно считать ее заготовкой на будущее, которая потребуется в следующем разделе 3.

Аксиома поведения $\boldsymbol{R}_{11}$ задает следующие правила запуска на счет любого математического объекта семейства родов структур «модель-компонента»:

Во-первых, считается, что в начале шага моделирования известны текущие элементы всех процессов и все внутренние характеристики модели (на первом шаге – это начальные значения внутренних характеристик и начальные элементы процессов).

Во-вторых, предполагается, наблюдаемость внешних характеристик модели в любой момент модельного времени.

Далее,

1. Вычисляются события связанные с текущими элементами процессов. Связь событий с текущими элементами процессов определяется правилами переключений (9). Вычисляться события могут параллельно, однако для продвижения

вычислительного процесса далее, следует дождаться завершения вычислений всех событий. Если есть наступившие события, проверяется, нет ли переходов к быстрым элементам, если они есть – выполняются быстрые элементы (они становятся текущими). Вычисляться они могут также параллельно, однако для продвижения вычислительного процесса далее, следует дождаться завершения вычислений всех быстрых элементов, затем возврат к началу п.1; если нет переходов к быстрым элементам – совершаются переходы к новым медленным элементам, затем возврат к началу п.1.

2. Если нет наступивших событий – из всех прогнозов событий выбирается ближайший $\Delta\tau$.
3. Если стандартный шаг моделирования Δt не превосходит прогнозируемого времени до ближайшего события, $\Delta t \le \Delta\tau$ – вычисляем текущие медленные элементы со стандартным шагом Δt. В противном случае – вычисляем их с шагом времени до ближайшего спрогнозированного события $\Delta\tau$. Медленные элементы также можно вычислять параллельно, с ожиданием завершения последнего.
4. Возвращаемся к началу п.1.

Таким образом, род структуры «модель-компонента» полностью определен.

3. Синтез модели-комплекса из моделей-компонент

Пусть теперь есть два математических объекта из семейства родов структур «модель-компонента» Σ_N и $\Sigma_{N'}$, с базисными множествами $X, M, E, \{M_j\}_{j=1}^{N}, \{E_j\}_{j=1}^{N}$ и $X', M', E', \{M'_j\}_{j=1}^{N'}, \{E'_j\}_{j=1}^{N'}$ соответственно. Определим модель-комплекс, составленную из этих двух моделей-компонент. Назовем базисными множествами комплекса следующий набор множеств:

$$X \cup X', M \cup M', E \cup E', \{M_j\}_{j=1}^{N+N'}, \{E_j\}_{j=1}^{N+N'};$$

где $\{M_j\}_{j=1}^{N+N'}$ и $\{E_j\}_{j=1}^{N+N'}$ обозначают $M_1,\dots,M_N,M'_1,\dots,M'_{N'}$ и $E_1,\dots,E_N,E'_1,\dots,E'_{N'}$ соответственно. Объединения множеств $M \cup M'$ и $E \cup E'$ понимаем в самом обычном смысле:

$$M \cup M' = \{m : (m \in M) \vee (m \in M')\},$$

$$E \cup E' = \{e : (e \in E) \vee (e \in E')\}.$$

Что касается множеств X и X', будем считать, что они не пересекаются, т.е., если даже $X \cap X' \neq \emptyset$, договоримся, что мы умеем различать характеристики из $X \cap X'$ по признаку их происхождения из X или X'. Таким образом, непустые элементы $X \cap X'$ входят в объединение $X \cup X'$ дважды, как $X \cap X' \subset X$ и $X \cap X' \subset X'$, и различаются. Содержательный смысл такого предположения – мы собрали вместе две модели-компоненты, но они никак не взаимодействуют между собой – каждая живет собственной жизнью, так, как если бы она была одна.

Покажем, что на базисных множествах комплекса можно задать род структуры $\Sigma_{N+N'}$ из семейства «модель-компонента». Будем проверять выполнение следующих соотношений типизации:

$$x \cup x' \subset X \cup X', \; a \cup a' \subset X \cup X', \tag{11}$$

$$s \cup s' \subset M \cup M', \; f \cup f' \subset M \cup M', \tag{12}$$

$$\{m_{j,real} \subset M_j \times (M \cup M')\}_{j=1}^{N+N'}, \tag{13}$$

$$\{e_{j,real} \subset E_j \times (E \cup E')\}_{j=1}^{N+N'}, \tag{14}$$

$$\left\{m_{j,in} \subset M_j \times \beta(X \cup X')\right\}_{j=1}^{N+N'}, \tag{15}$$

$$\left\{m_{j,out} \subset M_j \times \beta(X \cup X'\right\}_{j=1}^{N+N'}, \tag{16}$$

$$\left\{e_{j,in} \subset E_j \times \beta(X \cup X')\right\}_{j=1}^{N+N'}, \tag{17}$$

$$\left\{m_j^0 \subset M_j\right\}_{j=1}^{N+N'}, \tag{18}$$

$$\left\{sw_j \subset E_j \times M_j \times M_j\right\}_{j=1}^{N+N'}, \tag{19}$$

$$\left\{p_j \subset \beta(M_j) \times \beta(E_j) \times M_j \times \beta(E_j \times M_j \times (M \cup M')\right\}_{j=1}^{N+N'}; \tag{20}$$

и аксиом:

$$R_1:\ (x \cup x' \cup a \cup a' = X \cup X')\ \&\ ((x \cup x') \cap (a \cup a') = \varnothing),$$

$$R_2:\ (s \cup s' \cup f \cup f' = M \cup M')\ \&\ ((s \cup s') \cap (f \cup f') = \varnothing),$$

$$R_3:\ \left\{(\forall m \in M_j)(\exists!\tilde{m} \in M \cup M')(\{m, \tilde{m}\} \in m_{j,real})\right\}_{j=1}^{N+N'},$$

$$R_4:\ \left\{(\forall e \in E_j)(\exists!\tilde{e} \in E \cup E')(\{e, \tilde{e}\} \in e_{j,real})\right\}_{j=1}^{N+N'},$$

$$R_5:\ \left\{(\forall m \in M_j)(\exists!r \in \beta(X \cup X'))(\{m, r\} \in m_{j,in})\right\}_{j=1}^{N+N'},$$

$$R_6:\ \left\{(\forall m \in M_j)(\exists!r \in \beta(x \cup x'))(\{m, r\} \in m_{j,out})\right\}_{j=1}^{N+N'},$$

$$R_7:\ \left\{(\forall e \in E_j)(\exists!r \in \beta(X \cup X'))(\{e, r\} \in e_{j,in})\right\}_{j=1}^{N+N'},$$

$$\begin{aligned} R_8:\ &\left\{((\forall e \in E_j)(\exists!r \in M_j \times M_j)(\{e, r\} \in sw_j))\ \&\right.\\ &\left.\&\ ((\{e, r\} \in sw_j, \{\tilde{e}, \tilde{r}\} \in sw_j, r = \tilde{r}) \Rightarrow (e = \tilde{e}))\right\}_{j=1}^{N+N'} \end{aligned}$$

$R_9: \left\{p_j = \left\{M_j, E_j, m_j^0, sw_j\right\}\right\}_{j=1}^{N+N'}$,

R_{10}: аксиома однозначности вычисления характеристик модели-компоненты,

R_{11}: аксиома поведения модели-компоненты (организации имитационных вычислений).

Очевидно, соотношения типизации (11) – (20) и аксиомы R_1 – R_9 выполняются, поскольку как для компоненты Σ_N, так и для $\Sigma_{N'}$, выполняются соотношения (1) – (10) и соответствующие аксиомы R_1 – R_9.

Для примера, рассмотрим самые громоздкие соотношения типизации (20). Пусть сначала $1 \le j \le N$, тогда, в силу (10), справедливо:

$$p_j \subset \beta(M_j) \times \beta(E_j) \times M_j \times \beta(E_j \times M_j \times M) \subset$$
$$\subset \beta(M_j) \times \beta(E_j) \times M_j \times \beta(E_j \times M_j \times (M \cup M')).$$

Пусть теперь $N < j \le N + N'$. Тогда снова в силу (10) заключаем:

$$p'_j \subset \beta(M'_j) \times \beta(E'_j) \times M'_j \times \beta(E'_j \times M'_j \times M') \subset$$
$$\subset \beta(M'_j) \times \beta(E'_j) \times M'_j \times \beta(E'_j \times M'_j \times (M \cup M')).$$

Самый громоздкий набор аксиом R_8, равно как и R_9, проверяется совсем уж просто – это всего лишь объединение соответствующих наборов аксиом для объектов Σ_N и $\Sigma_{N'}$, поэтому для примера проверим аксиомы R_6.

Пусть $1 \le j \le N$, тогда, в силу выполнения R_6 для Σ_N, справедливо:

$$(\forall m \in M_j)(\exists !r \in \beta(x))(\{m, r\} \in m_{j,out}).$$

Если $r \in \beta(x)$, тем более $r \in \beta(x \cup x')$, поскольку

$$\beta(x) \subset \beta(x \cup x').$$

Следовательно, справедливо:

$$\left(\forall m \in M_j\right)\left(\exists ! r \in \beta(x \cup x')\right)\left(\{m, r\} \in m_{j,out}\right).$$

Аналогично, если $N < j \le N + N'$, тогда, в силу выполнения R_6 для $\Sigma_{N'}$, справедливо:

$$\left(\forall m' \in M'_j\right)\left(\exists ! r' \in \beta(x')\right)\left(\{m', r'\} \in m'_{j,out}\right).$$

Если $r' \in \beta(x')$, то тем более $r' \in \beta(x \cup x')$, поскольку

$$\beta(x') \subset \beta(x \cup x').$$

Следовательно, справедливо:

$$\left(\forall m' \in M'_j\right)\left(\exists ! r' \in \beta(x \cup x')\right)\left(\{m', r'\} \in m'_{j,out}\right),$$

что и завершает проверку выполнения аксиом R_6.

Из сказанного выше заключаем, что наш комплекс является математическим объектом семейства родов структур «модель-компонента» $\Sigma_{N+N'}$.

Аксиома поведения R_{11} по определению такова, что применима к любому объекту семейства «модель-компонента».

Обсудим теперь аксиому однозначности вычисления характеристик модели-компоненты R_{10}.

Поскольку мы «принудительно» решили, что

$$X \cap X' = \varnothing, \tag{21}$$

то однозначность вычисления характеристик следует из таковой, имевшей место для каждой из компонент. Правда, при этом наши компоненты никак не взаимодействуют друг с другом, – они просто формально объединены вместе.

Обратим внимание на то, что формального критерия выполнения условия (21) быть не может. Формально можно лишь добиться выполнения (21), всегда считая разными характеристики, первоначально принадлежащие разным компонентам, как это и было сделано. Определить, какие характеристики модели одинаковы, а какие отличаются, может лишь разработчик, исходя из своих представлений о ее предметной области. Тем не менее, если разработчик указал, что условия (21) нарушены и конкретизировал, в чем состоят эти нарушения, можно указать на ряд вполне формальных действий, которые следует предпринять, чтобы исправить ситуацию.

Отметим, что если бы мы понимали объединение характеристик комплекса самым обычным образом:

$$X \cup X' = \left\{ y : (y \in X) \vee (y \in X') \right\},$$

не различая «происхождение» объединенных характеристик, то, как не трудно видеть, соотношения типизации (11) – (20) и аксиомы R_1 – R_9 также были бы выполнены. При этом в случае $X \cap X' \neq \emptyset$, имело бы место взаимодействие между компонентами, за счет пересечения части их характеристик. Однако гарантировать однозначность вычисления объединенных характеристик (аксиома R_{10}), вообще говоря, было бы невозможно. Действительно, в этом случае уже никакие гипотезы о замкнутости не запрещают двум методам-элементам разных процессов (первоначально принадлежащих разным компонентам), одновременно изменять одну и ту же характеристику, если эта характеристика принадлежит $X \cap X'$.

Попробуем найти примирение двух приведенных выше крайностей. Будем считать, во-первых, что по-прежнему все элементы объединения $X \cup X'$ формально различимы, т.е., формально $X \cap X' = \emptyset$. Во-вторых, предположим, что с содержательной точки зрения, наоборот, некоторые характеристики компонент одинаковы (например, обе модели-компоненты пытаются дать прогноз погоды на завтра), т.е., с точки зрения семантики модели $X \cap X' \neq \emptyset$.

Начнем с самого простого. Пусть пересекаются внешние характеристики наших компонент, например, по характеристике $\tilde{a}$. Это означает, что внешняя характеристика $a_i = \tilde{a}$ первой модели и внешняя характеристика $a'_j = \tilde{a}$ второй, отражают один и тот же (иначе они бы различались) фактор внешнего по отношению к нашим моделям мира. Из общих представлений о единстве и объективности внешнего мира следует, что значения этих характеристик (конечно, правильно идентифицированные) должны совпадать, и поэтому, в модели-комплексе вполне достаточно одной из них. Исключим для определенности a' из $X \cup X'$. Тогда во всех соотношениях коммутации (15) и (17), куда входит a', следует заменить a' на a, что позволит этим соотношениям сохранить прежний смысл в модели-комплексе. Если кому-то соображения о «единстве и объективности мира» кажутся не слишком убедительными, – ниже для устранения пересечения внутренних характеристик компонент, будет предложен иной способ, который вполне может быть применен также и в данном случае.

Пусть теперь пересекается множество внутренних характеристик одной модели-компоненты с множеством внешних характеристик другой.

Например, пусть $x \cap a' = \tilde{y} = x_i = a'_j$. Это означает, что если во второй модели-компоненте характеристика a'_j не моделировалась, а являлась фактором внешнего по отношению к этой компоненте мира, то в модели-комплексе та же характеристика моделируется явно, так как явно моделируется в первой модели-компоненте внутренней ее характеристикой x_i . Стало быть, в модели-комплексе внешняя характеристика второй компоненты $\tilde{y} = a'_j$ становится внутренней характеристикой первой модели-комплекса $\tilde{y} = x_i$. Стало быть, из объединения $X \cup X'$ следует исключить a'_j, а во все соотношения коммутации из (15) и (17), куда входит $\tilde{y} = a'_j$, вместо него включить $\tilde{y} = x_i$.

Таким образом, в полном соответствии с шутливой программистской поговоркой: «Описанная ошибка становится новой полезной возможностью», можно не только объединять компоненты в комплекс, но и использовать в модели-комплексе то, что некоторые из компонент, быть может, моделируют нечто, что без них не умели другие, извлекая определенные преимущества из усложнения модели путем создания комплексов.

Пусть, наконец, пересекаются внутренние характеристики моделей-компонент. К сожалению, звучавшие в случае внешних характеристик соображения о единстве и объективности мира здесь не убеждают. Наоборот, всем хорошо известно, что могут быть десятки различающихся прогнозов погоды или курсов валют на завтра. Пусть внутренние характеристики двух моделей-компонент пересекаются по характеристике $\tilde{x} = x_1 = x_2$. Предлагается дополнить множество $X \cup X'$ характеристикой $\tilde{x}$. Вопрос вычисления значения этой характеристики поручить специальной модели-компоненте, внешними характеристиками которой будут x_1 и x_2, алгоритм ее вычисления по x_1 и x_2 – оставляется на усмотрение разработчика. В соотношениях коммутации входящих параметров – это (15) – (17), где присутствуют x_1 и x_2, они должны быть заменены на $\tilde{x}$.

Теперь основной результат данного подраздела можно сформулировать в виде следующего предложения:

Предложение 3.3.1.

Пусть имеются n математических объектов семейства родов структур $\Sigma_{N_1},\dots,\Sigma_{N_n}$ «модель-компонента» с базисными множествами

$$X_1, M_1, E_1, \{M_{1,j}\}_{j=1}^{N_1}, \{E_{1,j}\}_{j=1}^{N_1};\ \dots;\ X_n, M_n, E_n, \{M_{n,j}\}_{j=1}^{N_n}, \{E_{n,j}\}_{j=1}^{N_n}.$$

Тогда возможно распространение рода структуры $\Sigma_{N_1+...+N_n}$ семейства «модель-компонента» на базисные множества модели-комплекса, составленной из этих моделей-компонент:

$$X_1 \cup ... \cup X_n, M_1 \cup ... \cup M_n, E_1 \cup ... \cup E_n, \{M_j\}_{j=1}^{N_1+...+N_n}, \{E_j\}_{j=1}^{N_1+...+N_n},$$

при этом:

1. В объединение характеристик $X_1 \cup ... \cup X_n$ входят все характеристики компонент, т.е., формально различными считаются даже одинаковые в предметной области характеристики разных компонент. Объединения реализаций методов-элементов $M_1 \cup ... \cup M_n$ и методов-событий $E_1 \cup ... \cup E_n$ понимаются в обычном смысле.
2. Если множества внутренних характеристик моделей-компонент $x_1 \subset X_1, ..., x_n \subset X_n$ имеют непустые попарные пересечения, то исходный набор из n моделей-компонент придется пополнить еще некоторым количеством L объектов семейства родов структур «модель-компонента». Первоначальное множество характеристик комплекса также пополняется одной новой характеристикой, на каждую дополнительную компоненту.
3. Если попарно пересекаются внутренние и внешние характеристики компонент: $x_i \subset X_i$, $a_j \subset X_j, i \neq j$ и $x_i \cap a_j \neq \varnothing$, то эти характеристики коммутируются, при этом коммутируемые внешние характеристики компонент исключаются из набора внешних характеристик комплекса, так как становятся его внутренними характеристиками.
4. Если попарно пересекаются внешние характеристики моделей-компонент – они либо отождествляются, либо, как и в случае с внутренними характеристиками, неоднозначность разрешается путем добавления в комплекс

некоторого количества дополнительных компонент и характеристик. ■

Таким образом, модели-компоненты допускают объединение в комплексы, с возможной (но не всегда обязательной) коммутацией некоторых внутренних характеристик некоторых компонент с некоторыми внешними характеристиками других, возможным дополнением первоначального набора компонент некоторыми новыми и коррекцией части соответствий в соотношениях (5) – (7) для некоторых компонент. Как было показано, полученная таким образом модель-комплекс обладает структурой рода «модель-компонента» (быть может, после добавления в комплекс еще некоторого количества компонент, разрешающих возникшие при объединении базисных множеств неоднозначности), и, стало быть, в свою очередь может входить в новые модели-комплексы.

Этим способом можно строить сколь угодно сложные фрактальные конструкции комплексов, состоящих из компонент, которые сами затем становятся компонентами комплексов. При этом, тем не менее, все вычислительные процессы конструкций произвольной сложности, полностью определяются аксиомой поведения R_{11} семейства родов структур «модель-компонента».

4. Модельный синтез и объектный анализ

Теперь остановимся на сравнении основных понятий модельно-ориентированного и объектно-ориентированного программирования. Прежде всего, отметим, что объектно-ориентированный подход в настоящее время представлен в двух видах: один из них можно назвать «базовым», в его основе лежат такие понятия, как класс, объект, типизация, наследование, инкапсуляция, полиморфизм, которые реализованы с некоторыми нюансами в большинстве современных императивных языков программирования, таких как C++, Java, C#, Delphi и многих других. Второй, который можно назвать «расширенным», представлен унифицированным языком моделирования UML [14], [15]. Язык UML включает все перечисленные выше базовые понятия, кроме того, его создатели пошли по пути резкого увеличения числа исходных понятий и представлений. Например, кроме «вертикального» отношения наследования, которое в UML чаще называется отношением обобщения (при этом отношение обобщения направлено от потомка к предку), имеются отношения ассоциации, композиции, агрегации и зависимости. Появилась возможность описывать поведение систем, причем даже несколькими способами: диаграммы взаимодействия, диаграммы состояний и диаграммы деятельности. Вообще в UML очень многое можно делать несколькими способами, что делает его удобным инструментом знатока, но весьма затрудняет жизнь новичку. Сами авторы языка говорят, что: «UML подчиняется правилу 80/20, т.е., 80% большинства проблем можно смоделировать, используя 20% средств UML» [14].

Начнем с «базового» варианта. Такие понятия как класс, объект, типизация в двух рассматриваемых подходах понимаются примерно одинаково. Следует лишь заметить, что корни этих понятий следует искать конечно не в C++ и даже не в Симуле-67, а скорее в работах Н. Бурбаки, структуралистов XX века, вплоть до Эрлангенской программы Ф. Клейна.

Примерно одинаково понимаются также характеристики и методы. Что же касается использования методов – здесь имеется существенная разница. Она состоит в том, что в объектном программировании объект затем и нужен, чтобы вызывать его методы в различных программах, причем в качестве параметров методу можно передать, а также принять от него любые переменные, удовлетворяющие его сигнатуре – не обязательно характеристики объекта. В модельно-ориентированном программировании метод модели-компоненты может работать лишь с ее характеристиками, и вызывать его «вручную» нет ни возможности, ни необходимости – он будет вызван автоматически, когда это потребуется. Инкапсуляция в модельно-ориентированном программировании такова, что не позволяет напрямую добраться до методов. Оперировать можно лишь с моделями-компонентами.

Это вовсе не означает, что методы совсем недоступны. Наоборот, например, в реализованном макете распределенной системы имитационного моделирования [9], [10], библиотеки методов публикуются в Интернете для общего использования и возможны модели-компоненты, не имеющие ни одного локального метода, притом все реализации методов могут физически находиться в разных местах Сети. Тем не менее, воспользоваться методом можно лишь включив его в состав некоторой модели-компоненты – таков уровень инкапсуляции. В некоторых случаях это может показаться усложнением, но оно оплачено отсутствием забот об организации поведения модели-компоненты – она всегда ведет себя так, как умеет, и поэтому включение ее в любой комплекс всего лишь вопрос правильной коммутации.

Скажем несколько слов о наследовании. Отношение наследования для множества классов объектно-ориентированного языка программирования есть отношение частичного порядка. Классы, не имеющие предков, но обладающие потомками, называются по отношению к ним корневыми или базовыми. Классы, не имеющие потомков, называются листовыми.

Проектирование в объектно-ориентированной парадигме большой программной системы состоит в воплощении базовых понятий и представлений этой системы в базовые классы объектов и построении затем иерархии классов, развивающих, конкретизирующих и воплощающих эти идеи во множестве листовых классов, с помощью которых и будет строиться целевая программная система. Такой дедуктивный способ проектирования большой программной системы хорош, когда она творится «с чистого листа», как мир у Платона. В имитационном моделировании, однако, гораздо чаще возникает задача не создания новых миров, а воспроизведения фрагментов уже существующего. В подобном фрагменте запросто могут быть собраны вместе «...диван, чемодан, саквояж, картина, корзина, картонка и маленькая собачонка». Друг из друга они не выводятся, а восходить вверх до их общих предков – достаточно бессмысленно. Для подобных задач «базовому» варианту объектного подхода недостает наследования снизу вверх (в UML такое наследование имеется). В модельно-ориентированном программировании объединение моделей-компонент в модель-комплекс можно считать множественным наследованием снизу вверх.

Даже если в объектно-ориентированном проекте с помощью наследования построена самая замечательная иерархия классов, все равно все сложности организации вычислительного процесса, состоящего в использовании разработанных и отлаженных методов листовых классов, лежит на разработчике системы: чтобы система что-то делала – необходимо организовать вызов нужных методов в нужной последовательности. Самый сложный этап построения системы остается неформализованным – это искусство.

Попытки формализовать процесс проектирования сложных программных систем и породили UML. По-видимому, и в самом деле на UML можно описать любую систему, и даже с нескольких точек зрения. Вопрос в том, что делать дальше с таким описанием – здесь нет единства мнений. Некоторые специалисты (например, [15]) считают, что основная ценность UML как раз в применении как средства документирования и обмена формализованными описаниями на стадиях эскиза и проектирования сложных систем. Тем не менее, имеется и ряд средств, позволяющих компилировать UML-описания в заготовки классов универсальных языков программирования, и в этом случае можно говорить о режиме использования UML в качестве языка программирования. Однако здесь мы снова остаемся в рамках объектной парадигмы – получаем иерархию классов и заготовок классов, но не избавляемся от необходимости писать императивные программы, вызывающие в нужном порядке методы этих классов.

Заключение

Объединение моделей-компонент в модель-комплекс можно считать неким аналогом наследования, если сравнивать модельный синтез с объектным анализом. Только это наследование чаще всего множественное и происходит не сверху вниз, а наоборот, – снизу вверх. Такое объединение с точки зрения геометрической теории декомпозиции [5], может быть названо Р-композицией моделей-компонент. Р-композиция позволяет строить модели фрактальной структуры сколь угодно большой степени сложности. При этом построение разбивается на вполне обозримые этапы описания на декларативном языке компонент и комплексов модели, написание на универсальных языках программирования, но в функциональной парадигме, методов-элементов и методов-событий компонент, отладки отдельных компонент по принципу – отладил и забыл. Императивное программирование [12], при этом исключается из проекта, что облегчает его отладку.

В силу аксиомы R_{11} рода структуры «модель-компонента», вычислительный процесс всегда, даже для сверхсложной фрактальной модели, организуется автоматически универсальной, однажды написанной и отлаженной программой реализующей эту аксиому, единой для любых моделей-компонент. Более того, оказывается, что чем сложнее фрактальная конструкция модели – тем более высокая степень параллельности кода, который производит программа, реализующая аксиому R_{11} рода структуры «модель-компонента».

Благодарности. *Автор благодарен Юрию Николаевичу Павловскому за многолетние, но ненавязчивые попытки (без малейшей надежды на успех) приобщения его к аппарату родов структур Н. Бурбаки и геометрической теории декомпозиции.*

Список литературы

[1] *Бусленко Н.П.* Моделирование сложных систем М.: Наука, 1978, 400 с.
[2] *Бусленко Н.П.* Сложная система //Статья в Большой Советской Энциклопедии, 3-е изд., М.: Советская энциклопедия, 1969-1978.
[3] *Бурбаки Н.* Теория множеств. М.: Мир. 1965. 456 с.
[4] *Павловский Ю.Н.* Геометрическая теория декомпозиции и некоторые ее приложения. М.: ВЦ РАН, 2011, 93 с.

[5] *Павловский Ю.Н., Смирнова Т.Г.* Введение в геометрическую теорию декомпозиции. М.: ФАЗИС: ВЦ РАН, 2006, 169 с.

[6] *Павловский Ю.Н., Смирнова Т.Г.* Проблема декомпозиции в математическом моделировании. М.: ФАЗИС: 1998, 272 с.

[7] *Елкин В.И.* Редукция нелинейных управляемых систем. Декомпозиция и инвариантность по возмущениям. М.: ФАЗИС, 2003. 207 с.

[8] *Бродский Ю.И.* Модельный синтез и модельно-ориентированное программирование М.: ВЦ РАН, 2013, 141 с.

[9] *Бродский Ю.И.* Распределенное имитационное моделирование сложных систем М.: ВЦ РАН, 2010, 156 с.

[10] *Бродский Ю.И., Павловский Ю.Н.* Разработка инструментальной системы распределенного имитационного моделирования. //Информационные технологии и вычислительные системы, №4, 2009, С. 9-21.

[11] *Бродский Ю.И., Лебедев В.Ю.* Инструментальная система имитации MISS М.: ВЦ АН СССР, 1991, 180 с.

[12] *Бродский Ю.И., Мягков А.Н.* Декларативное и императивное программирование в имитационном моделировании сложных многокомпонентных систем //Вестник МГТУ им. Н.Э. Баумана. Сер. Естественные науки. Спец. выпуск № 4 «Математическое моделирование». - 2012. - С.178-187.

[13] *Пономарев И.Н.* Введение в математическую логику и роды структур: Учебное пособие. М.: МФТИ, 2007, 244 с.

[14] *Буч Г., Рамбо Д., Якобсон И.* Введение в UML от создателей языка. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2012, 494 с.

[15] *Фаулер М.* UML. Основы., 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 192 с.

Об авторе:

Юрий Игоревич Бродский

Ведущий научный сотрудник ВЦ РАН, автор более 150 научных работ. Область научных интересов – распределенное имитационное моделирование сложных систем, инструментальные средства моделирования, моделирование социальных процессов. В течение последних лет читал лекции по математическому и имитационному моделированию в МГТУ им. Н.Э. Баумана, МПГУ, МГУ им. М.В. Ломоносова.

e-mail: yury_brodsky@mail.ru

Образец ссылки на публикацию:

Ю.И. Бродский. Синтез многокомпонентных имитационных моделей и модельно-ориентированное программирование // Программные системы: теория и приложения: электрон. научн. журн. 2013. Т. 4, № 3(17), с. 1–33.

URL: http://psta.psiras.ru/read/???

Yu.I. Brodsky. Synthesis of multi-component simulation models and model-oriented programming.

Abstract. The work is devoted to the description and simulation of complex systems, about which it is well known of what components they are made, what those components are able to do, what rules of interaction they obey. The challenging problem of modeling is to reproduce the behavior and to evaluate the capabilities of such a system as a whole. A new approach to the design and implementation of computer simulation models of complex multicomponent systems is introduced. It differs from the last decades mainstream of the object-oriented approach. The central concept of this approach and at the same time, the basic building block for the construction of any more complex structures is the concept of the model components. Model component endowed with a more complicated structure than, for example, the object in the object-oriented analysis. This structure provides to the model-component an independent behavior - the ability of a standard way to respond to standard requests of its internal and external environment. At the same time, the computer implementation of model-component's behavior is invariant with respect to the integration of models-components into complexes, which allows firstly to build a fractal pattern of any complexity and secondly to implement a computational process of such structures uniformly - by a single program. In addition, the proposed paradigm of the multi-component simulations allows to exclude imperative programming code and to generate code with a high degree of parallelism.

Key Words and Phrases: Simulation, Complex Multi-Component Systems, Model-Oriented Programming, Parallel and Distributed Computing.