Вычислительные технологии, 2023, том 28, № 1, с. 6–22. © ФИЦ ИВТ, 2023
Computational Technologies, 2023, vol. 28, no. 1, pp. 6–22. © FRC ICT, 2023
ISSN 1560-7534
eISSN 2313-691X

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

DOI:10.25743/ICT.2023.28.1.002

Моделирование развивающихся систем с активными элементами

И. Н. Скопин[1,2]

[1]Институт вычислительной математики и математической геофизики СО РАН, 630090, Новосибирск, Россия

[2]Новосибирский государственный университет, 630090, Новосибирск, Россия

Контактный автор: Скопин Игорь Николаевич, e-mail: iskopin@gmail.com

Поступила 30 апреля 2022 г., доработана 20 декабря 2022 г., принята в печать 27 декабря 2022 г.

Предлагается подход к моделированию развивающихся систем, в основе которого лежит идея независимого построения взаимодействующих аспектных моделей за счет использования инструментальной вычислительной среды. Основные черты подхода заключаются в отказе от постулата о детерминированности поведения систем, развитие которых обеспечивается активностью элементов, и в событийном управлении их поведением. Обсуждаются основные требования к инструментальному обеспечению подхода.

Ключевые слова: развивающиеся системы, активный элемент, атрибутное представление, аспектная модель, моделирование, мультиаспектность, многомерность, множественное структурирование, инструментальная поддержка.

Цитирование: Скопин И.Н. Моделирование развивающихся систем с активными элементами. Вычислительные технологии. 2023; 28(1):6–22. DOI:10.25743/ICT.2023.28.1.002.

Введение

Изучение развивающихся систем, поведение которых существенно зависит от индивидуальной активности их элементов, является актуальным для широкого круга исследований. С практической точки зрения важно уметь управлять такими системами, чтобы создавать желаемые и ограничивать нежелательные условия развития. Но при обычно применяемых подходах к изучению и проектированию систем это достигается с трудом. Основная причина тому — направленность исследований на поиск общего детерминированного закона развития системы, тогда как для эффективного управления требуется не общий закон, а организация целенаправленной индивидуальной деятельности элементов системы. Однако если главным двигателем развития является индивидуальная деятельность элементов, то такой закон просто не существует.

Проблемой исследования развивающихся систем с активными элементами является учет поведения элементов не только в этой системе, но и в других важных аспектах. Эта многоаспектная активность может противоречить цели модельной системы и обычно рассматривается как внешнее неконтролируемое воздействие. Цели поведения элемента редко совпадают с целями содержащей его системы. Поэтому в модели системы в целом трудно отразить вовлеченность ее элементов во внешние по отношению к ней действия, изменяющие поведение элементов системы. Чтобы избежать сложности из-за различий

в поведении элементов, связанных с разными аспектами, обычно для них строят разные аспектные модели. При этом возникает не менее сложная проблема сопряжения моделей. Предлагаемый подход призван преодолеть как постулат детерминизма, так и трудности многоаспектного характера.

В качестве конечной цели исследования, предлагаемого в данной работе, рассматривается создание инструментария для поддержки изучения систем указанного типа. Основным препятствием на пути к ее достижению является необходимость преодоления стереотипов, сложившихся при использовании традиционных методов моделирования в исследовании развивающихся систем. И первое, что нужно сделать, это дать точные определения используемых понятий. Обсуждению этой методологической базы посвящено основное содержание нашей работы. Только на этой основе можно предлагать требования к средствам и инструментам поддержки моделирования с учетом особенностей конкретных областей исследования.

1. Анализ предшествующих исследований

Наш подход можно рассматривать как вариант дискретно-событийного моделирования поведения систем, предложенного еще в 1960-х гг. Г. Гордоном в языке GPSS [1]. В наиболее развитом виде дискретно-событийный подход представлен в методологии моделирования с использованием языков Simula и Simula 67 [2, 3].

Первые системы дискретно-событийного моделирования разрабатывались с учетом обеспечения вычислимости моделей только на однопроцессорных ЭВМ. Сегодня этот анахронизм привел бы к чрезмерному ограничению возможностей моделирования (см. ниже). Поэтому мы предлагаем отказаться от устаревшего управления вычислениями на основе глобально (линейно) упорядоченных событий, которое использовалось в упомянутой и других подобных системах главным образом для оптимизации вычислений, а также для представления глобального времени в модели. Последнее при изучении развития систем с активными элементами может привести к некорректным результатам моделирования (см. [4], где дано обоснование только что упомянутого понятия глобального времени при моделировании развивающихся систем). В связи с нашей концепцией уместно отметить обзор В. Окольнишникова [5], в котором обсуждается современное понимание времени в имитационном моделировании.

Еще одна особенность нашего подхода заключается в поддержке автономного моделирования аспектов, что роднит его с аспектно-ориентированным программированием [6]. Отличие нашего подхода от только что упомянутого в том, что мы предлагаем технологизировать поддержку аспектных моделей в рамках специальной среды проектирования, обеспечивающей стандартизированное представление модельных элементов.

Концептуально близкой нашему подходу является разработка мультиагентных систем (МАС) в том виде, в каком она определена в [7]. Один из российских исследователей и разработчиков в этой области П. Скобелев говорит: “Ключевым элементом систем является программный агент, способный воспринимать ситуацию, принимать решения и взаимодействовать с другими агентами. Эти возможности резко отличают МАС от существующих жестко организованных систем, придавая им такое важное новое свойство, как самоорганизация. В этом случае отдельные части программы могут договориться о том, как следует решать проблему. Части приобретают собственную активность и могут заранее инициировать диалог с пользователем в неустановленное время. Они могут работать в условиях неопределенности и предлагать уточнение и переформулировку

задач и т. д.” [8]. Отказ от нахождения детерминированного принципа, позволяющего выявить наилучшее решение, активность агентов с их целенаправленным поведением, ориентация на операции с развивающимися системами — общее понятие для обоих подходов.

Однако вне мультиагентного подхода остается проблема поддержки взаимного влияния аспектов. Информация, влияющая на поведение агента, поступает из его окружения и передается другим агентам для использования. Но это не самостоятельное построение взаимосвязанных аспектных моделей, поддерживаемых инструментально, что декларируется в нашем подходе как одно из основных требований. В то же время в нашем случае применимы традиционные методы построения мультиагентных систем, особенно при задании индивидуального поведения агентов. Следует отметить, что для предлагаемого подхода очень важным и критичным качеством поддержки является то, как организована среда функционирования агентов, а также какие вспомогательные инструменты предоставляются пользователям (в связи с этим в качестве примера хорошего решения можно указать на систему REPAST [9]). В следующих разделах мы представляем характерные особенности подхода и принципы, которые предлагаются для организации моделирования. Эта информация используется для обоснования основных требований к общему инструментальному обеспечению подхода. В заключении подводятся итоги и намечаются направления дальнейших исследований.

2. Особенности подхода

2.1. Модельная система, имитация и события

Мы говорим о *модели системы* как об абстрактном понятии, не привязываясь к какой-либо осмысленной интерпретации. Тем не менее, приступая к изучению *объекта*, всегда необходимо отделять его от остальной части реального мира, которую естественно рассматривать как *окружение объекта*. Уже это разделение естественно рассматривать как моделирование реальности: совершая его, исследователь абстрагируется от целостности мира. Дальнейшее построение модели заключается в выделении частей-элементов и, при необходимости, частей их окружений.

Выбирая части, необходимые для исследования, и игнорируя то, что кажется неважным, исследователь строит *модельную систему*. Она состоит из элементов, действия которых изменяют некоторые характеристики как самих элементов, так и системы в целом. Эта конструкция должна адекватно отражать поведение реальной системы. В модели характеристики представлены в виде атрибутов элементов и системы. Их изменение в процессе расчета можно интерпретировать как моделирование поведения или, что то же, симуляцию функционирования реальной системы.

Значение атрибута элемента изменяется только в результате выполнения действия программы, если в операционный контекст этой программы входит изменяемый атрибут. Постулируется, что все действия являются методами элементов (их классов — в смысле объектно-ориентированного программирования [10]), а операционные контексты формируются из всех доступных элементов и их атрибутов. Действие, т. е. метод, который выполняет элемент, — это проявление его активности в системе моделирования посредством модельных расчетов. Мы не накладываем ограничений на то, как описываются программы действий элементов, требуется лишь *единое унифицированное атрибутивное представление всех элементов* как общее основание описания действия.

В соответствии с целями данной работы мы должны определить механизмы, имитирующие развитие систем. В основе любого развития модели лежит обеспечение вариативности, т. е. изменчивости состояний ее элементов. В связи с этим декларируется наличие четырех основных состояний элементов:

- *Активные элементы* системы, обладающие способностью действовать, т. е. изменять свои свойства и свойства других элементов, в том числе иметь возможность порождать и уничтожать элементы, а также изменять свой статус (разумеется, с необходимыми ограничениями). Активный элемент проявляет свою активность при выполнении любого из действий, намеченных для него. Действие может быть *приостановлено* другим элементом, самим элементом или элементом из окружающей среды, в результате чего выполнение действия прекращается. Выполнение такого действия может быть продолжено за счет *реактивизации*, т. е. возобновления активности элемента другим элементом или элементом из его окружения.
- *Пассивные элементы* системы, свойства которых могут изменяться под влиянием активных элементов и/или окружения системы.
- *Порожденные элементы* системы. Они имеют состояние, указывающее на возможность стать компонентами системы (и получить статус пассивных или активных элементов).
- *Уничтоженные*, т. е. *разрушенные элементы* системы. Они перестали быть компонентами системы, хотя и остаются в ней как неизменяемые структуры данных.

При построении модельной системы исследователь должен предусмотреть способы передачи информации между элементами. Для этого предложено разрешить доступ элементов друг к другу с помощью ссылок. Ссылки как особый вид атрибутов элемента позволяют получать доступ к атрибутам других элементов. Это дает возможность использовать и изменять атрибуты (включая ссылки) и влиять на действия элементов, связанных ссылками. Элемент может порождать другие элементы, связанные с родительским элементом, добавлять связи (отношения) или удалять их в соответствии с требованиями моделирования. Он может удалить все свои связи с другими элементами, что интерпретируется как его уничтожение. Все эти действия выстраивают *сеть модельной системы*.

Удобно предположить, что система в целом представлена специальным элементом, указывающим на эту систему в модели. Он называется *индикаторным элементом системы* или *индикатором системы*. Этот элемент имеет собственное атрибутивное представление, что позволяет различать внешние и внутренние воздействия на атрибуты системы и на атрибуты ее элементов. Элемент-индикатор получает априорные родственные связи со всеми остальными элементами системы. Понятно, что этот элемент не обязательно должен иметь прототип в реальной системе.

Любое действие модельной системы запускается как реакция на какое-либо событие, происходящее в течение моделирования. *Событийный механизм*, используемый в предлагаемом подходе, отражает трактовку Хоара, согласно которой любое изменение в модельной системе является событием, если имеются элементы, *распознающие* его. Если распознавание события предусмотрено для элемента, то выполняется одно из его действий [11]. Выполнение этого действия называется *реакцией на событие*.

Типы всех событий перечислены заранее, но набор типов событий, на которые может реагировать элемент, формируется в динамике вычислений (при выполнении действия того или иного элемента). Мы считаем, что текущий набор распознаваемых типов событий элемента задается как специальный атрибут, кодирующий набор типов событий.

Зная значение этого атрибута, всегда имеем возможность найти все типы событий, которые может распознать элемент. В правильно построенном атрибутивном представлении верно и обратное. Если это условие не выполняется, считается, что нарушена целостность атрибутивного представления элемента.

Реакция на общее событие, распознаваемое несколькими элементами, выполняется *совместно* и *асинхронно*. Это позволяет выполнять действия параллельно (разумеется, с известными ограничениями). Реакции выполняются как обычные параллельные процессы в собственных потоках, они синхронизируются с помощью обычных средств согласования выполнения процессов. Следуя Хоару, мы считаем, что распознавание события является мгновенным действием, а при выполнении реакции возможно возникновение других событий, на которые должен реагировать элемент.

При использовании событийного механизма вполне естественно определяется понятие *протокола поведения элемента* как последовательность троек:

$$(< \text{состояние элемента} >, < \text{событие, которое он распознает} >, < \text{реакция элемента} >)\,.$$

Это позволяет корректно определить *поведение модельной системы* как множество всех протоколов элементов со склеенными событиями, совместно распознаваемыми разными элементами. Эта конструкция задает частичный порядок всех произошедших событий, где каждый протокол представляет собой линейную цепь. Протокол элемента (точнее, последовательность его событий) естественно рассматривать как *локальное время элемента*, и тогда указанный частичный порядок на всех событиях трактуется как корректно *определенное глобальное время модельной системы*. Важное уточнение: под корректностью мы понимаем наиболее точную информацию об отношениях между событиями: *“раньше”*, *“позже”*, *“неизвестно, когда”*. В нашей концепции модельного времени отсутствует *“приблизительная одновременность”*, характерная для повседневного употребления времени. Такая “аппроксимация” по существу приводит к неправильному детерминизму с произвольно определенными временными отношениями. Заметим, что мы не исключаем возможности уточнения такого “приближения”, но всегда настаиваем на использовании только достоверной информации. Наше понимание времени существенно отличается от того, которое используется в системах с дискретными событиями и в мультиагентных системах. Подробности о локальном и глобальном времени можно найти в статье [4].

2.2. Определение эволюционирующих (развивающихся) систем

Понятия системы и развивающейся системы используются очень широко и, к сожалению, в весьма разных смыслах. По этой причине при разработке инструментария поддержки моделирования таких систем необходимо определиться с терминологией. Для однозначного понимания, о каких системах идет речь далее, в предыдущих разделах мы ввели ряд основных определений, которые, как будет показано, позволят достаточно точно представить понятия, связанные с развитием модельных систем, а в дальнейшем базовые требования к инструментарию поддержки их изучения. Мы определяем модельную систему как *развивающуюся* или *эволюционирующую*, если:

- Определены ее *состояния*. Они являются интегральной характеристикой системы, изменяющейся в результате действия активных элементов и/или получения соответствующих сигналов из внешней среды.

- Состояния системы могут быть представлены вершинами специальной графовой структуры, дуги которой определяют возможные переходы между состояниями. В каждый момент времени одно из состояний считается *текущим*, каждому переходу соответствует *условие срабатывания*: в текущем состоянии один или несколько переходов должны быть истинными, один из них ведет к *новому текущему состоянию* (правила выбора перехода должны определяться для конкретной модели).

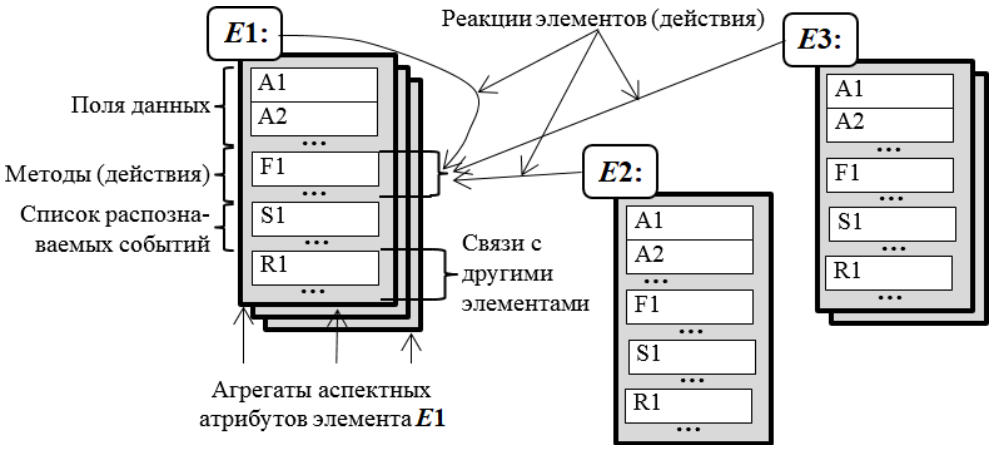
Последовательность текущих состояний системы, которая устанавливается при однократном расчете модели, называется *траекторией развития системы*. Совокупность всех траекторий, каждая из которых правильна в смысле некоторого критерия, называется *поведением модельной системы*. Мотивация использования только что представленного определения моделирования развивающихся систем становится понятной, если сопоставить его с классификационными признаками систем разных типов:

- *Статическая система* характеризуется следующим свойством — множество ее состояний фиксировано и не меняется при выполнении расчетов.
- *Для динамической системы* характерна изменчивость множества состояний. Поэтому в таких системах должны быть определены действия создания и устранения состояний, а также переходы между ними.
- *Детерминированная система* характеризуется *свойством постоянного поведения*: при любом состоянии и фиксированных возможных влияниях среды одни и те же действия элементов и влияний среды приводят к одним и тем же результатам.
- В *недетерминированной системе* есть состояния, в которых свойство постоянного поведения не выполняется, и, таким образом, постулируется, что эти системы не подчиняются общим законам развития.

Управлять развитием системы можно, изменяя входные параметры как в начале работы, так и при переходе системы из состояния в состояние. Такое управление в детерминированной модельной системе, адекватной реальному объекту исследования, имеет хорошие шансы на успех. В то же время в недетерминированной системе, развивающейся за счет активности ее элементов, параметрическое управление оказывается чрезмерно громоздким. В этом случае уместно пытаться оптимизировать разработку управляющих воздействий для достижения определенных целей. Вполне понятно, что управление без целей развития не имеет смысла, но, как мы уже упоминали выше, цели системы и ее элементов обычно не только не совпадают, а часто еще и противоречат друг другу. В результате остается один самый естественный способ управления: создание условий для поддержки положительных и подавления отрицательных устремлений элементов системы.

2.3. Многоаспектная структура системы и атрибутивное представление элементов

Поведение элементов в реальной системе предполагает возможность рассмотрения их деятельности в различных аспектах функционирования. На модельном уровне для описания этого можно строить специальные модели всех аспектов, признаваемых важными. Они называются *аспектными моделями*. Однако из-за сложности координации моделей обычно строится только общая модельная система, а аспекты игнорируются. В нашем подходе мы предлагаем поддержку независимого моделирования аспектов



Атрибутное представление элементов $E1$ (детализированное), $E2$ и $E3$ (краткое)
Attribute representation of elements $E1$ (in details), $E2$, and $E3$ (briefly)

и динамики взаимного влияния моделей аспектов в имитационных расчетах, которые являются общими для всех или нескольких из этих моделей.

Аспектная модель определяет структуру множества входящих в нее элементов и их отношений, называемую *аспектной структурой*. Это часть сети модельной системы. Модельную систему можно рассматривать как объединение всех аспектных структур. Когда модели аспектов создаются как совместно сосуществующие компоненты модельной системы с общими аспектными структурами (мы называем это *множественно структурированной системой* [14]), для реализации модельных вычислений можно кооперировать все или некоторые модели аспектов (в том числе и ограничиваться рассмотрением одной модели). Это становится достижимым, если предусмотреть специальный унифицированный формат атрибутивного представления элементов. По своей сути унификация сводится к соглашению о том, что атрибутивное представление разбито на блоки, относящиеся к каждой из видовых структур. Такие блоки называются *агрегатами аспектных атрибутов*. Каждый агрегат содержит данные, присвоенные этой структуре, текущее состояние, а также методы аспектных действий. Вся информация о любой аспектной модели распределяется по совокупности базовых элементов.

На схеме (см. рисунок) представлено соединение трех элементов системы. $E1$ участвует в трех аспектных моделях, которые показаны в виде трех перекрывающихся прямоугольников. $E2$ принимает участие только в одной аспектной модели (нет перекрывающихся прямоугольников). $E3$ участвует в двух аспектных моделях. $E1$ распознает свои события, а также события элементов $E2$ и $E3$. Это показано линиями со стрелками, указывающими на $E1$, реагирующий на распознанное событие.

Распознавание события и реакция элемента на него носят локальный характер, т. е. представлены в моделях определенных аспектов. Возможна реакция элемента на одно событие в нескольких моделях, но это означает лишь возникновение нескольких независимых реакций.

2.4. Горизонтальное и вертикальное моделирование системы

В качестве модели системы рассматривается совокупность всех аспектных моделей, совместное моделирование которых формирует характеристики системы, представленные среди атрибутов индикаторного элемента системы. Для проверки различных гипотез,

касающихся управления моделируемой системой, можно использовать усечение совокупности совместных аспектов. В этом случае для расчета выбираются не все аспектные модели, а влияние отсутствующих аспектов на их поведение задается извне. Такое оперирование аспектными моделями называется *горизонтальным моделированием системы.*

С точки зрения горизонтального моделирования реализация поведения аспектных моделей не важна. Нужно только, чтобы в необходимых ситуациях аспектные модели генерировали события, обеспечивающие адекватность поведения в других аспектах, а элемент в каждой аспектной модели правильно реагировал на внешние события. В частности, аспектная модель может быть основана на принципах, отличных от соглашений нашего подхода. При этом поведение элементов в этой модели должно соответствовать их поведению в любых других аспектных моделях. В противном случае придется констатировать концептуальную несовместимость подходов. При выполнении условия совместимости может быть поставлена задача замены аспектуальных моделей реальными данными из внешних источников. Это переводит нашу концепцию на уровень систем реального времени для принятия управленческих решений. Опуская подробности, отметим, что широкое использование подобного подхода в практических приложениях мультиагентных систем (см., например, [8]) свидетельствует о его эффективности и в нашем случае.

Соглашение о едином формате атрибутивных представлений элементов в информации о системе в целом (как представление индикаторного элемента системы) позволяет моделировать поведение системы так, что ее можно использовать как элемент какой-либо *надсистемы.* Наша система будет являться обычным активным элементом надсистемы в горизонтальной модели более высокого уровня.

Подсистемы как поставщики агрегированной информации для надсистем следующих уровней могут предоставлять данные моделирования или информацию о реальном поведении прототипов подсистем. Для удовлетворения разных потребностей можно строить разные надсистемы, которые, в свою очередь, будут использоваться как активные элементы еще более высоких уровней моделей. Таким образом, многоуровневое моделирование может быть источником данных для иерархического управления в различных областях. Такое построение *называется вертикальным моделированием системы.* Как и в случае с горизонтальным моделированием, этот подход позволяет заменять модели реальными данными.

2.5. Группы элементов, подсистемы и операции над группами

В модельной системе можно выделить *содержательно связанные группы элементов*, которые иногда удобно рассматривать как независимые. Совокупность атрибутивных описаний элементов таких групп часто называют *подсистемой* модели, но это не совсем точно: система (и подсистема) не сводится к сумме ее частей. Необходимо говорить о собственных атрибутах подсистемы, т. е. подсистемы получают статус элементов, влияющих на поведение системы в целом. Этот формальный статус можно рассматривать как определение: индикаторный элемент группы (т. е. связанный со всеми элементами группы специально заданным отношением, называемым *группировкой*) и все элементы группы называются *подсистемой.* Отношение группировки является динамическим, поскольку принадлежность элемента к группе, а также его связь с индикаторным элементом подсистемы могут изменяться.

Атрибуты подсистемы формируются как за счет собственного поведения (действий ее элементов), так и под влиянием поведения других элементов, обеспечивающих подсистему внешней информацией. В частности, связи, возникающие в подсистеме как отношения с внешними элементами, задаются для подсистемы в целом, и она может перенаправлять их на свои элементы. В то же время это не исключает возможности самостоятельного распознавания событий элементами подсистем.

На множестве, состоящем из групп, в том числе и групп, состоящих из всех элементов каждой аспектной структуры, определяются *теоретико-множественные операции: пересечение, объединение и дополнение*. Их результаты можно рассматривать как группы. Иногда их интерпретируют с точки зрения их собственного поведения.

Формально можно ввести операцию, называемую *сверткой подсистемы*. Ее выполнение заключается в замене взаимодействия элементов подсистемы с внешней средой взаимодействием индикаторного элемента группы (подсистемы). При этом генерация событий для них передается от элементов подсистемы к ее индикаторному элементу. Свернутую подсистему можно рассматривать как черный ящик, входы которого предоставляют информацию, произведенную реакциями подсистемы на события, а выходы генерируют события из подсистемы, предоставляя информацию для окружающей среды. Но такое рассмотрение не совсем корректно: всякая подсистема, формируясь в каком-то аспекте, не может отгораживаться от своих элементов, принадлежащих другим аспектным моделям.

Свертка вполне применима и при нисходящем построении моделей. В этом подходе конструирование модельной системы начинается с верхнего уровня, который задает наиболее содержательные аспекты моделирования, указывая элементы подсистем и отношения между элементами. Далее определяются события и реакции подсистем на них. Определение атрибутивных описаний подсистем-элементов завершает построение верхнего уровня. Следующим шагом является раскрытие подсистем, т. е. формирование их групп элементов и распределение выстроенного аспектного поведения среди вновь созданных элементов. Это действие выполняется для всех существенных аспектов. Может потребоваться формирование новых аспектов и их распространение на всю систему. Рассмотрение подсистемы как черного ящика позволяет реализовать альтернативные раскрытия подсистем. Это может включать моделирование, не связанное с нашим подходом, но требует гармонизации подходов в смысле свертки. Раскрытие подсистем на следующих уровнях аналогично.

3. Иллюстративный пример

Для наглядного представления предлагаемого подхода в данном разделе представлен фрагмент решения условной задачи, которая при должной разработке (уточнение, наполнение содержанием и т. д.) могла бы стать основой реализации реального процесса управления. Мы не ставим перед собой такой задачи и ограничиваемся демонстрацией только того, что улучшит понимание нашего предложения.

Пусть *директор* предприятия получил *уведомление* о том, что в связи с эпидемией можно организовать вакцинацию сотрудников (здесь и далее курсивом выделены роли лиц, задействованных в проведении вакцинации, и мероприятия, связанные с ней). Директор назначает *ответственное лицо*, которому поручает подготовить материалы к технико-экономическому обоснованию вакцинации: ее необходимость и возможность осуществления. Если условия приемлемы, ответственному лицу поручаются *закупка*

вакцины и другие действия, возникающие в рамках организации вакцинации. Он выполняет эти действия совместно с *секретарем*, подготавливающим объявления, резервирование помещений и т. д. Вакцинация заканчивается *отчетом* ответственного лица и освобождением его от соответствующих обязанностей.

Фрагмент модели (см. таблицу), относящийся к решению задачи, в виде конечно-автоматной диаграммы показывает поведение двух исполнителей в двух аспектах — *директора* и *ответственного*. Принятые обозначения пояснений не требуют. Следует лишь отметить, что генератор событий (конструкция **G**) позволяет задавать реагирующие элементы, а для формирования сообщений используется параметризация событий. Действия на диаграмме представлены неформально. Для их реального описания может использоваться любой объектно-ориентированный язык со специальными библиотеками поддержки взаимосвязей аспектов как при подготовке моделей, так и при формировании отчетов.

Директор в аспекте *деятельностей* может работать с различными событиями, влияющими на производство. На своем предприятии ему доступны атрибутивные представления всех работников. Используя эту информацию, можно выделить кандидатуру ответственного лица непосредственно. Проверяя, например, путем моделирования различных лиц, которым может быть поручено задание, директор выбирает того, кто показывает наиболее качественные результаты в следующих действиях: обоснованно выбрать медицинскую услугу, составить более содержательный отчет, уложиться в приемлемые сроки (задача требует учета того, что на уровне системной модели и, возможно, нескольких аспектных моделей должно быть отражено время, так как это существенный неуправляемый фактор распространения эпидемии).

Ответственное лицо в аспекте *вакцинации* использует модели, отражающие его деятельность. Для описания бухгалтерских данных вполне пригодны известные экономические модели, аспект распространения эпидемии приемлемо моделируется с помощью известных статистических методов и/или имитационных симуляций, снабженных генераторами событий, которые моделируют процесс заболевания работников как воздействие на производственные аспекты моделируемой системы. Из диаграммы видно следующее априорное решение: действие *Информация получена* (Inf_B, Inf_E) объединяет события *готовности информации* как от бухгалтерии, так и от эпидемической службы. Из этого следует, что элемент **Re** может выполнять свои действия только тогда, когда происходят оба события готовности. Отчетная деятельность требует специального пояснения. Что может служить образом отчета на уровне модели? Это, конечно, не настоящий документ, а некий атрибут элемента **Re**, характеризующий степень готовности. Невозможно определить, как формируется его значение в рассматриваемых аспектах. Единственное требование к моделям в отношении таких атрибутов, как готовность, качество работы и т. п., состоит в том, что существуют процессы, которые вычисляют их значения перед использованием. Такими процессами могут быть последовательности действий элементов с соответствующими результатами или внешние воздействия на расчеты модели.

Приведенный фрагмент решения управленческой задачи не претендует на достоверность и полноту организации мероприятий по вакцинации работников предприятия. Его единственная задача — дать представление об одном из методов работы с аспектными моделями, чтобы понять требования к инструментарию, обсуждаемые в следующем разделе.

Фрагмент решения управленческой задачи по организации вакцинации работников предприятия
A fragment of the solution for the managerial problem of organizing the vaccination of employees of the enterprise

Состояние	События	Действия	Новое состояние	Комментарий
Директор **аспект** = *деятельности*				
Ожидание	*Уведомление об эпидемии*	**Re** = *ответственное лицо* **G**: *Назначение вакцинации* / **Re**	Ожидание готовности	Выбор компетентного **Re**
	...	...	...	Другие события
Ожидание готовности	*Подготовка завершена* (*средства*)	**If** Проверка (*средства*)		Принятие решения о деятельности
		true→**G**: Приобретение вакцины / **Re** (*средства*) **G**: Проведение вакцинации / **Re** (*средства*)	Вакансия заполняется	
		false→**G**: отказ от вакцинации	Завершение	
Вакцинация проводится	*Готовность к вакцинации*	**G**: Задача вакцинации / **Se** // *secretary* //, **Re**	Завершение	Действия для **Se** не предусматриваются
Завершение	*Отчет готов* (**Re**)	Освобождение *ответственного лица*	Ожидание	
Ответственное лицо **аспект** = вакцинация				
Ожидание	*Назначение вакцинации*	**G**: Запрос эпидемической информации **G**: Бухгалтерский запрос	Ожидание информации	Генерация информации для других аспектов
Ожидание информации	*Информация получена* (Inf_E, Inf_B)	Расчет эффективности (Inf_E, Inf_B) m = определить выделяемые средства **G**: *Подготовка закончена* (m)	Подготовка закончена	Ожидание двух событий
Подготовка закончена	*Отказ от вакцинации*	Подготовка отчета об отказе **G**: Отчет готов / **Di**	Ожидание	**Di** — Директор
	Закупка вакцины (*средства*)	Закупка вакцины **G**: Отчет готов / **Di** Готовность к вакцинации	Вакцинация закончена	
Вакцинация закончена	*Проведение вакцинации*	**G**: Получение работы / **Se** (объявление) Мониторинг *деятельностей* **G**: Отчет готов / **Di**	Ожидание	

4. Первичные требования к инструментарию

Наш подход к изучению или разработке систем требует создания специального инструментария поддержки, который включает три категории средств:

- обеспечения динамики моделирования;
- управления имитациями, сбором и обработкой результатов моделирования;
- поддержки разработки моделей.

Имитационные расчеты каждой аспектной модели могут быть реализованы с использованием различных алгоритмов, в том числе внешних для обсуждаемого инструментария. Поэтому выдвигается общее требование для всех трех категорий: необходимо, чтобы инструментарий конструировался как открытая для расширения программная система с интерфейсом, который является единственным регулятором, контролирующим совместимость с внешними расчетными модулями.

4.1. Обеспечение динамики моделирования

Активность элементов и автономность их поведения в каждой аспектной модели, а также взаимовлияние моделей требуют, чтобы каждый элемент определялся как отдельный вычислительный процесс, выполняющийся в отдельном потоке. Каждый процесс имеет локальную память, которая заполняется для каждого элемента данными его атрибутивного представления. В нашем подходе метод, управляемый событиями, не требует общей памяти различных процессов. Если разделяемая память все-таки необходима, то для этого можно предусмотреть вспомогательные элементы, процессы которых отвечают за доставку информации базовым процессам по запросам. Понятно, что это не исключает возможности оптимизации основного механизма на системном уровне. В частности, в статически вычисляемых случаях операций с событиями возможна замена реакций прямым вызовом соответствующих методов (этот прием оптимизации обсуждается в разд. 3.5 работы [13]).

Наиболее полную передачу информации между процессами, отвечающую принципам, заявленным в разд. 2, обеспечивает механизм сообщений, связанных с событиями. С помощью сообщений можно передавать определенные данные, а также ссылки на атрибутивные представления различных элементов. Таким образом, существует возможность прямого воздействия элементов друг на друга. Такие эффекты, вообще говоря, могут оказаться некорректными, если не позаботиться о согласованной синхронизации. Для этого на уровне атрибутивных представлений должна быть построена система доступа, включающая и отключающая флаги, динамические приоритеты и другие известные функции поддержки взаимодействия процессов [11].

Таким образом реализуются параллельные асинхронные процессы, возникающие при моделировании. При необходимости согласование и синхронизация совместных реакций контролируются обычными методами, используемыми при реализации параллельных вычислений. Специфика моделирования требует, чтобы количество потоков, участвующих в моделировании, было достаточно большим. Поэтому для использования нашего инструментария необходимы высокопроизводительные аппаратные архитектуры. С другой стороны, по крайней мере для первых версий системы в качестве технической и программной среды моделирования целесообразно выбрать существующую универсальную систему с параллельными вычислениями. Поэтому имеет смысл развивать наш инструментарий как конкретное наращивание выбранной системы. Перспек-

тивной в этом отношении является система [9], пользователям которой предоставляется возможность адаптации алгоритмов к конкретным условиям работы.

4.2. Управление имитациями, сбор и обработка результатов моделирования

Управление имитациями используется для того, чтобы пользователь мог влиять на расчеты с целью выяснения значимости факторов моделей. Такое управление рассматривается как часть событийно-ориентированной техники. Мы требуем, чтобы все элементы распознавали особое событие, называемое *приостановкой* симуляции, и реагировали на него, сохраняя свой статус для возобновления или перезапуска симуляций в будущем. Приостановки позволяют получать текущую информацию о расчетах, изменять некоторые параметры моделей, добавлять или удалять элементы и задавать их взаимосвязи. В момент приостановки всем элементам присваивается особое состояние, в котором они реагируют на единственное *событие возобновления имитации*.

Посредством серии приостановок может быть организован *мониторинг поведения системы*, т. е. получение информации об изменении структуры, протоколов элементов и текущих значений интегральных характеристик. Эта информация собирается для обработки результатов. Для сравнения версий моделирования предусмотрены варианты сохранения приостановленных конфигураций моделируемой системы в репозитории. Стандартные средства контроля версий (см., например, [13]) должны быть дополнены возможностью выбора опций по интегральным характеристикам системы и ее элементов.

В нашем инструментарии нет необходимости разрабатывать специальные средства обработки результатов моделирования, поскольку на сегодня рынок программного обеспечения предлагает для этой цели множество передовых продуктов (см., например, [15]). Хорошим решением будет сосредоточиться на особенностях, связанных с возможностями упомянутой выше системы поддержки параллелизма в фрагментарном программировании [14].

4.3. Средства поддержки разработки моделей

Наш инструментарий должен предоставлять инструменты для создания моделей, редактирования, мониторинга и других операций с достаточно автономными частями моделей. Наиболее важные из них перечислены ниже.

- Редакторы для добавления и удаления элементов моделей, изменения их атрибутов и т. д.
- Верификаторы промежуточных и конечных результатов построения элементов.
- Валидаторы результатов построения элементов и подсистем.
- Статические (используются при проектировании) и динамические (используются при проведении модельных расчетов) инструменты поддержки целостности системы.
- Валидаторы целостности начальной конфигурации модельной системы и/или ее модификации во время приостановок и возобновлений, а также в ходе имитационного расчета и его окончания. Нужна проверка целостности моделей, которая запрещает неправильную настройку симуляции. В частности, начальная конфигурация должна быть реалистичной, т. е. достижимой из начального состояния сеанса моделирования. Если алгоритмическая проверка моделирования невозмож-

на, инструментарий должен обеспечивать поддержку действий пользователя для ручного управления конфигурациями.

Моделирование и поддержка имитационных расчетов должны опираться на удобные средства визуализации структур. Различные структуры множества элементов должны быть согласованы. При проектировании модели и мониторинге имитаций каждый из элементов должен быть показан совместно во всех своих аспектных структурах. Для правильного принятия решений выбора между удалением и сохранением элементов в разных построениях требуется обоснование. Возможности визуализации процесса моделирования должны поддерживать корректность таких построений.

Набор средств разработки для разных стратегий моделирования должен обеспечивать разные стратегии проектирования. В частности, мы предлагаем поддержку стратегий “снизу вверх” и “сверху вниз”.

При восходящей стратегии построение каждой аспектной модели начинается с первичного набора базовых элементов. Для каждого типа элементов предусмотрен определенный шаблон представления атрибутов. В этих шаблонах методы действий элементов представляются в виде заготовок: пустых подпрограмм, готовых к доработке аспектных моделей в будущем. Также необходимо обеспечивать возможность спецификации ссылок для определения свойств элементов как отношений и разрешения доступа одного элемента к другим. В этом случае могут потребоваться дополнительные типы элементов, например, для реализации формообразующих подсистем.

Разработка аспектной модели при нисходящей стратегии начинается с подсистем и их элементов, указывающих на подсистемы. Подсистемы дополняются новыми базовыми элементами или наборами уже готовых элементов. Новые или уточненные существующие элементы описываются так же, как и при восходящей стратегии.

В обоих случаях разработка моделей должна быть подкреплена средствами отслеживания жизненных циклов элементов, отражающими степень готовности их атрибутивных представлений. Жизненный цикл класса элементов считается завершенным при наличии пути от класса до его готовности к использованию для генерации подготовленных элементов, а также выполнении всех условий корректности описаний каждой аспектной модели.

После того как аспектные модели будут построены, согласовываются влияния действий этих моделей на поведение других аспектов. Результатом этого процесса могут быть идентификация классов, уточнение программ аспектных моделей и другие модификации описаний классов, направленные на обеспечение корректности горизонтальной модели системы.

Только что вкратце представленные действия пользователя при разработке моделей предполагают расширенную поддержку интерфейса. Необходимым требованием к этой поддержке является функциональная завершенность интерфейса, т. е. отражение в интерфейсе всех методов деятельности разработчика, связанных с созданием моделей [16].

Заключение

Предлагаемый подход к изучению и разработке систем направлен на поддержку построения программных моделей и выполнения серий имитационных расчетов. Результаты таких расчетов должны предоставить исследователю информацию о реальной системе. Для этого необходима обсуждаемая инструментальная поддержка подхода.

В настоящей работе мы не останавливались на вопросах внешнего ввода данных, вывода информации и лишь частично обсудили механизм контроля приостановок имитаций. Для реального использования модельных комплексов упомянутые вопросы безусловно важны, и они должны решаться на этапе проектирования. В принципе эти проблемы решаемы, но представляется целесообразным дать по ним конкретные предложения на основе детального анализа приведенных выше требований.

Завершая изложение, следует упомянуть о вычислительной сложности моделирования, которая связана с нашим подходом, применительно к практическим задачам. Многопоточное выполнение действий элементов, событийно-ориентированная техника и т. д. требуют больших вычислительных затрат. В связи с этим наш проект должен предусматривать использование существующего программного обеспечения общего назначения, а также разработку специальных средств оптимизации, зависящей от архитектуры. Для повышения производительности моделирования следует использовать адаптивную программную платформу. Хотя эта задача выходит за рамки нашего обсуждения, отметим, что ее необходимо решать в более общем контексте, подразумевающем организацию параллельных вычислений. В связи с этим разумным решением представляется использование упомянутой выше системы [9].

Благодарности. Работа выполнена по государственному контракту Министерства науки и высшего образования Российской Федерации с Институтом вычислительной математики и математической геофизики СО РАН (0251-2021-0005).

Список литературы

[1] **Liskov B.** GPSS session. History of programming languages. N.Y.: Academic Press; 1981: 403–437. DOI:10.1016/B978-0-12-745040-7.50013-2.

[2] **Dahl O.J., Myhrhaug B., Nygaard K.** SIMULA 67 common base language. Oslo: Norwegian Computer Centre; 1968: 145.

[3] **Dahl O.J., Nygaard K.** SIMULA — A language for programming and description of discrete event systems. Oslo: Norwegian Computer Centre; 1967: 127.

[4] **Скопин И.Н.** Локальное и глобальное время при моделировании развивающихся систем. Новосибирск: Проблемы информатики; 2020: (4):5–26. DOI:10.24411/2073-0667-2020-10013.

[5] **Окольнишников В.В.** Представление времени в имитационном моделировании. Вычислительные технологии. 2005; 10(5):57–80.

[6] **Kiczales G., Lamping J., Mendhekar A., Lopes C., Loingtier J.-M., Irwin J.** Aspect-oriented programming. Proceedings of the European Conference on Object-Oriented Programming (ECOOP). Finland, Springer-Verlag. LNCS. 1997; (1241):220–242.

[7] **Wooldridge M.** An introduction to multi-agent systems. 2nd edition. Wiley; 2009: 484. ISBN:978-0-470-51946-2.

[8] **Skobelev P., Budaev D., Laruhin V., Levin E.** Practical approach and multi-agent platform for designing real time adaptive scheduling systems. Communications in Computer and Information Science. Springer Verlag; 2014: 1–12. DOI:10.1007/978-3-319-07767-3_1. (In Russ.)

[9] REPAST: recursive porous agent simulation toolkit. Available at: `http://repast.sourceforge.net`.

[10] **Booch G.** Object-oriented analysis and design with applications. 2nd edition. Addison-Wesley; 2007: 717. ISBN:0-201-89551-X.

[11] **Hoare C.A.R.** Communicating sequential processes. Prentice-Hall; 1985: 260. ISBN:0-13153289-83317.

[12] **Скопин И.Н.** Иерархичность и моделирование развивающихся систем. Новосибирск: Проблемы системной информатики: Сб. науч. тр. / Под ред. В.Н. Касьянова. Новосибирск: Институт систем информатики им. А.П. Ершова СО РАН; 2010: 188–214.

[13] **Непейвода Н.Н., Скопин И.Н.** Основания программирования. М.; Ижевск: Институт компьютерных исследований; 2003: 851.

[14] **Malyshkin V., Sorokin S., Chayuk K.** Fragmentation of numerical algorithms for the Parallel Subroutine Library. LNCS. 2009; (5698):331–343.

[15] **Маклаков С.** CASE-инструменты для разработки информационных систем. М.: Диалог-МИПР; 2001: 340.

[16] **Скопин И.Н.** Разработка интерфейсов программных систем. Системная информатика. Проблемы архитектуры, анализа и разработки программных систем. Новосибирск: Наука; 1997; (6):34–96.

Вычислительные технологии, 2023, том 28, № 1, с. 6–22. © ФИЦ ИВТ, 2023 ISSN 1560-7534
Computational Technologies, 2023, vol. 28, no. 1, pp. 6–22. © FRC ICT, 2023 eISSN 2313-691X

MATHEMATICAL MODELLING

DOI:10.25743/ICT.2023.28.1.002

Simulation of developing systems with active elements

I. N. Skopin[1,2]

[1]Institute of Computational Mathematics and Mathematical Geophysics SB RAS, 630090, Novosibirsk, Russia
[2]Novosibirsk State University, 630090, Novosibirsk, Russia
Corresponding author: Skopin Igor N., e-mail: iskopin@gmail.com
Received April 30, 2022, revised December 20, 2022, accepted December 27, 2022.

Abstract

An approach to modelling of the developing systems based on the idea of independent construction of aspect models is proposed. The models are interacting through instrumental computing environment. Essentially, the postulate of the deterministic behaviour of systems is relaxed in the approach, the development of which is ensured by the activity of elements and event-driven control of their behaviour. The basic requirements for instrumental support of the approach are discussed.

The proposed approach for studying and developing systems is aimed at supporting the construction of software models and fulfilment of series of simulations. The results of simulations should provide the information about the real system. This requires an instrumental support of the approach to ensure the development of models, carrying out calculations performance control, management of simulations, collecting and processing of results.

In this paper, we do not elaborate on issues of the external data input, output of information, but briefly discuss the outlined mechanism for monitoring the simulation pauses. These are important points for the real use of model complexes and they should be solved at the next stage of the design. In principle, indicated problems are solvable, but it seems reasonable to provide concrete proposals for these issues based on a detailed analysis of requirements given above.

We should mention the issues of computational complexity of simulation, which are associated with our approach as applied to practical problems. Multi-thread execution of the elements actions, event-driven technique, etc. requires high computational cost. In this regard, our project should incorporate the use of existing general purpose software combined with a development of special facilities of the architecture dependent optimization. To improve the performance of the simulation, an adaptive software platform is required. Although this issue is beyond the scope of our discussion, we note that it must be solved in a more general context, implying the organization of parallel computations. In this connection, the use of the above-mentioned system seems to be a reasonable solution.

Keywords: developing systems, active element, attributive representation, aspect model, simulation, multi-aspect, multidimensional nature, multiple structure, tool support.

Citation: Skopin I.N. Simulation of developing systems with active elements. Computational Technologies. 2023; 28(1):6–22. DOI:10.25743/ICT.2023.28.1.002. (In Russ.)

Acknowledgements. This work was carried out under state contract of the Ministry of Science and Higher Education of the Russian Federation with The Institute of Computational Mathematics and Mathematical Geophysics SB RAS (0251-2021-0005).

References

1. **Liskov B.** GPSS session. History of programming languages. N.Y.: Academic Press; 1981: 403–437. DOI:10.1016/B978-0-12-745040-7.50013-2.
2. **Dahl O.J., Myhrhaug B., Nygaard K.** SIMULA 67 common base language. Oslo: Norwegian Computer Centre; 1968: 145.
3. **Dahl O.J., Nygaard K.** SIMULA — A language for programming and description of discrete event systems. Oslo: Norwegian Computer Centre; 1967: 127.
4. **Skopin I.** Lokal'noe I global'noe vremya pri modelirovanii razvivayushchikhsya sistem [Local and global time in modelling developing systems]. Problems of Informatics. 2020; (4):5–26. DOI:10.24411/2073-0667-2020-10013. (In Russ.)
5. **Okolnishnikov V.** Time representation in imitational modelling. Computational Technologies. 2005; 10(5):57–80. (In Russ.)
6. **Kiczales G., Lamping J., Mendhekar A., Lopes C., Loingtier J.-M., Irwin J.** Aspect-oriented programming. Proceedings of the European Conference on Object-Oriented Programming (ECOOP). Finland, Springer-Verlag. LNCS. 1997; (1241):220–242.
7. **Wooldridge M.** An introduction to multi-agent systems. 2nd edition. Wiley; 2009: 484. ISBN:978-0-470-51946-2.
8. **Skobelev P., Budaev D., Laruhin V., Levin E.** Practical approach and multi-agent platform for designing real time adaptive scheduling systems. Communications in Computer and Information Science. Springer Verlag; 2014: 1–12. DOI:10.1007/978-3-319-07767-3_1.(In Russ.)
9. REPAST: recursive porous agent simulation toolkit. Available at: `http://repast.sourceforge.net`.
10. **Booch G.** Object-oriented analysis and design with applications. 2nd edition. Addison-Wesley; 2007: 717. ISBN:0-201-89551-X.
11. **Hoare C.A.R.** Communicating sequential processes. Prentice-Hall; 1985: 260. ISBN:0-13153289-83317.
12. **Skopin I.** Hierarchy and developing simulation systems [Ierarkhichnost' i modelirovanie razvivayushchikhsya sistem]. Novosibirsk: Problems of System Informatics. Ltd. "Siberian Scientific Publishers"; 2010: 188–214. (In Russ.)
13. **Nepeivoda N., Skopin I.** Osnovaniya programmirovaniya [Foundation of programming]. Moscow; Izhevsk: Institute of Computer Sciences Researching; 2003: 915. (In Russ.)
14. **Malyshkin V., Sorokin S., Chayuk K.** Fragmentation of numerical algorithms for the Parallel Subroutine Library. LNCS. 2009; (5698):331–343.
15. **Maklalov S.** BPwin i Erwin. CASE-sredstva dlya razrabotki informatsionnykh sistem [BPwin and Erwin. CASE-tools of developing information systems]. Moscow: Dialog-MIFI; 2001: 340. (In Russ.)
16. **Skopin I.** Razrabotka interfeysov programmnykh sistem [Development of software systems interfaces]. System Informatics, issue 6 "The problems of architecture, analysis and software development". Novosibirsk: Nauka; 1997; (6):34–96. (In Russ.)