Разработка инструментальной системы распределенного имитационного моделирования[1]

Ю.И. Бродский, Ю.Н. Павловский

Аннотация. Одним их направлений развития имитационного моделирования является переход от разработки отдельных моделей к разработке их систем [1]. В работе описывается концепция инструментальной системы распределенного моделирования, позволяющей объединять имитационные модели, находящиеся на разных компьютерах, в распределенные модельные комплексы. Обсуждаются вопросы выбора архитектуры инструментальной системы; способы взаимодействия распределенных частей модели; концепция моделирования, а также набор средств, предоставляемых проектируемой системой для воплощения этой концепции.

Ключевые слова: имитация, сложные системы, декомпозиция, концепции моделирования, распределенное моделирование, инструментальные средства распределенного моделирования.

1. Компоненты и комплексы

1.1. Компонента - основная составляющая распределенной модели

Основными понятиями данной работы будут понятия комплекса и компоненты. Это связанные между собой понятия, способные переходить друг в друга. Так, комплекс, внутренняя структура которого сложна и образована множеством компонент, извне может восприниматься как единая компонента. В то же время, любая из компонент может обладать внутренней структурой, образующей комплекс.

Ближайшие два раздела посвящены объяснению, что есть в предлагаемой концепции компонента. Основной абстракцией, которой мы будем оперировать далее, будет динамическая система

$$\vec{x}_{i+1} = \vec{F}(\vec{x}_i, \vec{a}_i, \Delta t), \qquad (1)$$

где вектор $\vec{x}_i$ - набор значений внутренних или фазовых характеристик системы, вектор $\vec{a}_i$ - набор ее внешних характеристик или параметров в момент времени t_i : $\vec{x}_i = \vec{x}\ (t_i)$, $\vec{a}_i = \vec{a}_i\ (t_i)$, $\Delta t = t_{i+1} - t_i$ - интервал модельного времени, $\vec{F}$ - алгоритм, детерминированный или стохастический, позволяющий для любого i по значениям $\vec{x}_i$, $\vec{a}_i$ и Δt определять реализацию фазовых характеристик $\vec{x}_{i+1}$ на следующем шаге динамического процесса. Существование такого алгоритма означает замкнутость системы: зная начальное состояние системы $\vec{x}_0$ и значения $\vec{a}_i$, $i = 0,1,\ldots,n$, внешних характеристик, можно рассчитать внутренние характеристики системы в любой момент модельного времени t_i, $i = 0,1,\ldots,n$. Предполагается, что интервал времени, на котором нас интересует процесс, описываемый

[1] Работа выполнена при финансовой поддержке РФФИ, грант 07-07-00071-а.

(1), задан и равен $(t_0, t_0 + \Delta t \cdot n)$. Динамическую систему (1) будем называть моделью. В тех случаях, когда нужно подчеркнуть ее вхождение в качестве компоненты в более крупную модель, – компонентой или подмоделью, а в тех случаях, когда нужно подчеркнуть ее собственную многокомпонентную внутреннюю структуру, – комплексом или метамоделью.

1.2. Реакция компоненты на события

Известно, что реальные сложные системы, являющиеся предметом моделирования, в зависимости от определенных сочетаний своих характеристик и характеристик окружающего мира, способны порой достаточно резко менять свое поведение. Для отражения этого их свойства введем формализованное определение события. Будем считать, что события могут происходить в самом начале интервала модельного времени Δt, при переходе системы от момента t_i к моменту t_{i+1}. С моделью может быть связано несколько событий. Событие $k, k = 1,...K$ состоит в том, что некая скалярная функция $\Phi_k(\vec{x}, \vec{a})$ обращается в нуль на наборе характеристик модели $\vec{x}_i$, $\vec{a}_i$ в начале очередного интервала моделирования: $\Phi_k(\vec{x}_i, \vec{a}_i) = 0$. Будем считать, что имеется некоторое количество способов поведения модели, задающееся различными алгоритмами $\vec{f}_m(\vec{x}, \vec{a}, \Delta t), m = 1,..., M$, и однозначное отображение множества 2^K всевозможных реализаций всех связанных с моделью событий, во множество M различных алгоритмов ее поведения, которое и задает реакцию модели на произошедшее событие.

Заметим, что, начав с системно-динамического описания компоненты модели, мы пришли к двойственному, объектно-ориентированному описанию: компонента, определенная выше, задает класс объектов со свойствами $\vec{x}_i$, $\vec{a}_i$, методами $\vec{f}_m(\vec{x}, \vec{a}, \Delta t), m = 1,..., M$, переключения между методами задаются событиями $\Phi_k(\vec{x}_i, \vec{a}_i) = 0$, $k = 1,...K$.

1.3. Анализ комплекса (метамодели)

Как говорилось выше, мы предполагаем замкнутость модели (1), т.е., некоторую ее изолированность от внешнего мира, состоящую в том, что, зная векторы $\vec{x}_0$ и $\vec{a}_i$, $i = 0,1,...,n$, мы можем рассчитать траекторию системы на любом шаге модельного времени. На самом же деле все в этом мире связано, в том числе характеристики $\vec{x}_i$ и $\vec{a}_i$. Вопрос лишь в том, на каких характерных временах эти связи столь существенны, что их нужно учитывать, имея в виду необходимую с практической точки зрения точность, с которой дается прогноз значений внутренних характеристик. Предположение о замкнутости модели (1) означает, что на временном интервале $(t_0, t_0 + \Delta t \cdot n)$ характеристики $\vec{a}_i$ с точностью, определяемой практическими соображениями, не зависят от $\vec{x}_i$. Увеличение точности и глубины прогнозов, доставляемых существующими и эксплуатируемыми имитационными моделями, может привести к необходимости учитывать эту зависимость. Наиболее общей является ситуация? когда $\vec{a}_i$ и $\vec{x}_i$ взаимосвязаны. Это означает, что характеристики $\vec{a}_i$ также определяются динамической системой, в которой внешними характеристиками являются $\vec{x}_i$:

$$\vec{a}_{i+1} = \vec{\Psi}(\vec{a}_i, \vec{x}_i, \Delta t). \qquad (2)$$

Системы (1) и (2) можно, конечно, рассматривать как единую систему. Однако, во-первых, компьютерная реализация такой системы нерациональна, поскольку из предыдущего текста вытекает, что рассматривается ситуация, когда взаимное влияние $\vec{a}_i$ и $\vec{x}_i$ нужно учитывать не на каждом шаге Δt, а реже, в соответствии с соотношением характерных времен, определяющих изменение значений характеристик рассматриваемых процессов. Во-вторых, и это самое главное, может случить-

ся, что над моделью $\vec{a}_{i+1} = \vec{\Psi}(\vec{a}_i, \vec{x}_i, \Delta t)$ уже давно и успешно работает некий другой исследователь, для которого характеристики $\vec{a}_i$ являются внутренними, а $\vec{x}_i$ - внешними и которому для повышения глубины и точности прогноза также стало необходимо учитывать взаимное влияние $\vec{a}_i$ и $\vec{x}_i$. Возникает задача объединения уже существующих и эксплуатируемых моделей (1) и (2) с наименьшими затратами как человеческих, так и машинных ресурсов. Ситуация, когда взаимосвязанными из-за необходимости повышать глубину и точность прогноза оказываются только два процесса, является простейшей. В следующем разделе рассматривается ситуация, когда необходимо учитывать взаимное влияние друг на друга не двух процессов, а их произвольного количества.

1.4. Синтез комплекса (метамодели)

Пусть имеется N процессов, описываемыми динамическими системами $\vec{x}^k{}_{i+1} = \vec{F}^k(\vec{x}^k{}_i, \vec{a}^k{}_i, \Delta t)$, $k = 1,..., N$. Эти системы будут трактоваться как подмодели единого процесса, называться компонентами и обозначаться A_k, $k = 1,..., N$. Рассматривается ситуация, когда при необходимой точности и глубине прогноза компоненты уже нельзя рассматривать как независимые друг от друга. В то же время для достижения необходимой точности достаточно учесть факт зависимости некоторых из внешних характеристик некоторых компонент от некоторых внутренних характеристик некоторых других компонент. При этом мы располагаем информацией, необходимой для такого учета.

Для каждой из $N(N-1)$ пар $(A_i, A_j), i \neq j$, можно построить матрицу коммутации $Q_{i,j}$ размера $n_i \times m_j$, где n_i - размерность вектора $\vec{a}^i$ компоненты A_i, а m_j - размерность вектора $\vec{x}^j$ компоненты A_j. На пересечении p-й строки ($1 \le p \le n_i$) и q-го столбца ($1 \le q \le m_j$) этой матрицы стоит 1, если внешняя переменная a^i_p компоненты A_i зависит от внутренней переменной x^j_q компоненты A_j и 0 – в противном случае. Задание матриц $Q_{i,j}$, $1 \le i, j \le N; i \neq j$, которые будут называться матрицами коммутации, полностью решает вопрос информационного синтеза комплекса из компонент. Здесь следует заметить, что при фиксированном i и при $1 \le j \le N; i \neq j$, наличие единицы в фиксированной строке более чем в одной из матриц $Q_{i,j}$, так же как и наличие в одной строке более одной единицы, свидетельствовало бы о моделировании одной величины несколькими способами и, следовательно, ставило бы вопрос о согласованности таких моделей. Стало быть, подобная ситуация должна вызывать если не сообщение о фатальной ошибке при коммутации компонент комплекса, то, по крайней мере, очень серьезное предупреждение с призывом тщательно проверить согласованность компонент. С другой стороны, наличие нескольких единиц в столбце матрицы $Q_{i,j}$, а также, при фиксированном j и при $1 \le i \le N; i \neq j$, наличие единицы в фиксированном столбце более чем одной из матриц $Q_{i,j}$ говорит о том, что внутренняя переменная одной из компонент используется в качестве внешней переменной более чем одной компоненты, что не противоречит нашей концепции и должно быть разрешено. Следует также заметить, что поскольку единица, как следует из сказанного выше, – достаточно «редкий гость» в матрицах коммутации, то при описании комплекса на формальном языке будет проще специальным оператором языка описывать только существующие коммутации вместо выписывания всех матриц коммутации, которые должны автоматически строиться и проверяться на этапе компиляции описания комплекса.

Обсудим теперь вопрос о синхронизации во времени работы компонент (подмоделей). Предлагается следующий механизм синхронизации модельного времени компонент комплекса.

- Из соображений выбора масштаба и точности моделирования определяется «стандартный» интервал моделирования Δt.

- На интервале моделирования Δt постулируется механизм аппроксимации значений векторов $\vec{x}, \vec{a}$, позволяющий по известным их значениям $(\vec{x}_i, \vec{a}_i)$ и $(\vec{x}_{i+1}, \vec{a}_{i+1})$ на концах этого интервала вычислить их значения в любой промежуточной точке интервала. Это может быть линейная или какая-либо другая аппроксимация. Выбор того или иного способа аппроксимации зависит от масштаба моделирования и желаемой точности модели. Зачем это нужно - выяснится в следующем пункте.
- Для каждого события каждой подмодели ищется решение t уравнения $\Phi_k(\vec{x}(t), \vec{a}(t)) = 0$, $k = 1,...K$ на интервале Δt. Если таких решений нет, в качестве следующего интервала моделирования снова выбирается стандартный интервал моделирования Δt, если же решения есть, то наименьшее из них будет правым концом следующего интервала моделирования. Таким образом, интервал моделирования может уменьшаться с тем, чтобы ни одна из подмоделей не «проскочила» очередное событие, которое может существенно повлиять на ее поведение. (Необходимо избегать так называемой «дурной бесконечности» событий, руководствуясь принципом достаточности конечного числа событий на любом конечном интервале времени, сформулированном в [2], т.е. точек накопления событий на конечном интервале времени не должно быть.)

В заключение данного раздела следует отметить, что комплекс, изнутри состоящий из многих компонент, вовне может проявляться в качестве единой компоненты. Введем следующую операцию объединения компонент комплекса.

1. Внутренними переменными комплекса объявляется объединение внутренних переменных всех его компонент.
2. Методами комплекса объявляется объединение всех методов его компонент.
3. Событиями комплекса объявляется объединение всех событий его компонент.
4. Внешними переменными комплекса объявляется объединение всех внешних переменных его компонент, из которого исключаются все те переменные a_p^i, для которых p-я строка одной из матриц коммутации $Q_{i,j}$, $1 \le i, j \le N; i \ne j$, содержит единицу.

Мы видим, что тогда как изнутри комплекс имеет сложный многокомпонентный состав, извне он, в соответствии с данным в начале работы определением, вполне может быть воспринят как единая компонента. Единственное отличие от данного выше определения компоненты – возможность параллельного (т.е. одновременного) выполнения нескольких методов. Это отличие можно снять, например, разрешив компоненте одновременно выполнять несколько потоков (процессов), состоящих в чередовании некоторого числа выполняемых последовательно методов, как это было в MISS [2]. Так или иначе, определяющим остается то что, задавая для комплекса, рассматриваемого как единая компонента, начальные данные и значения внешних переменных, можно однозначно определить его внутренние переменные на любом шаге моделирования.

Таким образом, любая модель может быть представлена, с одной стороны, агрегировано, как единая компонента, а с другой стороны, - подробно, причем с произвольной степенью подробности, как комплекс компонент, в котором каждая из компонент в свою очередь допускает представление в виде комплекса, тем самым позволяя реализовывать концепцию мультимоделирования, т.е. построения семейства моделей разной степени подробности для изучения одного и того же явления.

2. Архитектура инструментальной системы распределенного имитационного моделирования

2.1. Глобальная сеть имитационного моделирования и ее функции

Система распределенного имитационного моделирования представляется пиринговой [например, http://www.ip2p.ru/articles/] сетью средней степени централизации.

Несколько десятков серверов глобальной сети имитационного моделирования должны выполнять следующие функции обслуживания рабочих станций (количество рабочих станций зависит от

количества компьютеров, на которых содержатся модели компонент, предназначенных для генерации распределенного моделирующего комплекса; при достаточном развитии предлагаемой технологии распределенного имитационного моделирования их может быть очень много - до нескольких тысяч).

- Регистрация клиентских рабочих станций и предоставляемых ими в сеть разделяемых ресурсов, т.е. моделей, компонент моделей, данных, алгоритмов и т.д.
- Хранение и постоянное поддержание баз данных модельных ресурсов в сети, содержащих информацию о том, какими рабочими станциями предоставлены в сеть те или иные модели, компоненты моделей и прочие ресурсы; какие из рабочих станций реально присутствуют в сети, какая из них и когда была доступна в последний раз, возможно, расписаний присутствия рабочих станций в сети.
- Поиск ресурсов в сети, т.е. уже существующих моделей, компонент моделей, необходимых данных и алгоритмов (примерно так, как осуществляется поиск различных файлов в современных пиринговых файлообменных сетях).
- Осуществление функций посредников стандартных запросов рабочих станций.
- Осуществление некоторых организационных функций, например, возможности нескольким рабочим станциям договориться о проведении совместной сессии имитационных экспериментов определенной продолжительности, с началом в определенное время, или же спланировать время проведения такой сессии в соответствии с их расписаниями присутствия в сети.

2.2. Программное обеспечение рабочих станций

Клиентское программное обеспечение рабочих станций такой сети вполне могло бы базироваться на описанных выше принципах анализа и синтеза модели. А именно, основу программного обеспечения рабочей станции составляет «система программирования» на некотором непроцедурном языке описания моделей и их компонент. Эта система программирования должна включать редактор (желательно и с графическим интерфейсом) описания моделей и их компонент. При описании компонент модели той или иной конструкции должна быть возможность ссылки на найденные уже существующие в сети компоненты, подходящие на роль компонент описываемой конструкции. Далее, система программирования клиентской части должна включать средства отладки описаний отдельных компонент и всей модели в целом. Далее, в клиентской части должен присутствовать сборщик модели, создающий базу характеристик компонент модели, проверяя при этом ее целостность.

Блок клиентской части инструментальной системы, ответственный за организацию имитационных экспериментов, должен с помощью существующих интеграционных технологий вызывать как локальные, так и удаленные компоненты модели по их известным адресам, передавая им необходимые характеристики из базы данных модели и принимая от них обратно в базу обработанные характеристики. Служебные методы, предоставляемые системой, также должны быть рассчитаны как на локальное, так и на удаленное использование. На этапе имитационного эксперимента рабочая станция, проводящая эксперимент, работает напрямую, минуя серверы, с рабочими станциями, содержащими используемые в модели компоненты. Как и в современных пиринговых сетях, серверы здесь нужны лишь на этапе поиска партнеров.

Уже отлаженные работающие модели и их компоненты могут быть объявлены доступными для всеобщего использования и пополнить фонд доступных в сети моделей и их компонент. При этом автоматически становятся доступными также и описания модели и ее компонент на языке описаний, по которым можно судить о ее адекватности тем или иным способам применения в других моделях. Таким образом, каждая рабочая станция может предоставлять в общее пользование как свой вычислительный потенциал, так и существующие наработки в области моделирования.

3. Язык описания комплексов и компонент

3.1. Определение типов записей фазовых переменных, параметров и констант

В данном разделе разобран специализированный непроцедурный язык, на котором должны составляться спецификации классов компонент модели. Отметим, что хотя аккуратнее употреблять термин "спецификация класса компонент", мы ради краткости будем пользоваться словосочетанием "спецификация компоненты". Лингвистические формулы записываются ниже в стандартной нотации, согласно которой служебные слова выделяются жирным шрифтом, разделители - двойными кавычками, а необязательные включения квадратными или фигурными скобками, причем фигурные скобки указывают на возможность повторения. Разбор языка начнем с двух нетривиальных конструкций, применяемых в спецификациях всех типов, каковыми являются определение структуры данных и оператор коммутации.

Собирая модель, инструментальная система автоматически генерирует ее базу данных, руководствуясь при этом содержимым специальных параграфов спецификаций компонент. Речь идет о параграфах определений типов фазовых переменных и констант. Их синтаксис различается только открывающей ключевой конструкцией, а в остальном сводится к правилам определения структур данных. Формально определение структуры данных есть набор определений списков полей, т.е. набор списков идентификаторов, завершающихся указаниями типов поименованных величин. При этом допускаются простые типы, структурные типы и тип массива. К простым относятся встроенные и каталогизированные типы, фиксируемые своими идентификаторами. Структурные типы - это типы обычных записей, ключей виртуальных записей и ключей виртуальных списков. Их указания выглядят как выделенные служебными словами наборы определений списков полей. Наконец, диапазоны индексов допускаются лишь числовые и тип элемента массива должен быть простым или структурным типом.

В стандартной нотации определение структуры данных выглядит следующим образом:

ОпрСтруктДанных: **::=** ОпрГруппыПолей { ОпрГруппыПолей }.ОпрГруппыПолей **::=**
Имя { "." Имя } ":" УказаниеТипа .
УказаниеТипа **::=** ИмяПростогоТипа ";" |
УказаниеСтруктурногоТипа |УказаниеТипаМассива
|УказаниеТипаПеречисления.
ИмяПростогоТипа **::=** ИмяВстроенногоТипа |
ИмяКаталогизированногоТипа.

ИмяВстроенногоТипа **::=** BOOLEAN | CHAR | BYTE | CARDINAL |BITSET | INTEGER |
LONGINT | REAL |
INDEFINITE | TEXT | NAME | PICTURE |BACKGRND.

ИмяКаталогизированногоТипа **::=** Имя

УказаниеСтруктурногоТипа **::=** ИмяСтруктурногоТипа
ОпрСтруктДанных END ";".

ИмяСтруктурногоТипа **::=** RECORD | LIST OF | BLOCKS OF |ADDRESS OF.
УказаниеТипаМассива **::=** ARRAY Диапазон { "," Диапазон
}OF ТипЭлемента.

Диапазон :: = " [" ЦелоеЧисло ".." ЦелоеЧисло "]".

ТипЭлемента **::=** ИмяПростогоТипа ";" |
УказаниеСтруктурногоТипа.

УказаниеТипаПеречисления :: = " (" Имя { "," Имя } ");".

В этих формулах термин Имя означает любую последовательность символов без пробелов длиной не более 20, не содержащую использованных в контексте разделителей.

Понятие каталогизированного типа и специфичные встроенные типы INDEFINITE, TEXT, NAME, PICTURE, BACKGRND, LIST OF, BLOCKS OF, ADDRESS OF будут разобраны ниже. Прочие встроенные типы определяются следующей таблицей соответствий:

Тип в инструментальной системе	Тип в МОДУЛЕ-2	Тип в Языке С	Длина в байтах
BOOLEAN	BOOLEAN	Char	1
CHAR	CHAR	Char	1
BYTE	BYTE	Char	1
перечисление	перечисление		1
BITSET	BITSET	Unsigned	2
CARDINAL	CARDINAL	Unsigned.	2
INTEGER	INTEGER	int	2
LONGINT	LONGINT	long	4
REAL	LONGREAL	double	8

Чтобы была возможность использовать в разных спецификациях одни и те же структурные типы данных, введен аппарат каталогизированных типов: в библиотеке системы, наряду со спецификациями групп, объектов и приборов, можно заводить в специально выделенной секции спецификации структурных типов данных, тем самым каталогизируя их. Синтаксис спецификации каталогизированного типа таков:

ОписательКаталогТипа. :: = **TYPE** Имя ";"
ОпрСтруктДанных **END ";".**

Смысл употребленных в правой части терминов тот же, что в предыдущей серии определений, причем Имя - и есть то имя, под которым данный тип будет зарегистрирован в библиотеке и под которым его можно упоминать в других спецификациях. Отметим, что приведенное определение рекурсивно, т.е. допускает использование в спецификации каталогизированного типа имен иных типов того же сорта. Циклов в этой рекурсии быть не должно, и система не допустит их, отказываясь компилировать спецификацию, если еще не заведены или не откомпилированы спецификации всех упоминаемых в ней каталогизированных типов.

Примеры спецификаций каталогизированных типов:

```
TYPE VECTOR;
        X, Y, Z : REAL;
     END;
```

3.2. Операторы коммутации

Согласно принятой концепции, часть параметров компонент комплекса может явно моделироваться в других его компонентах, т. е. являться частью фазовых переменных этих компонент. Комплексу также разрешено иметь собственные параметры и фазовые переменные. Для описания возможных связей параметров с фазовыми переменными используются операторы коммутации. Реально стыковка параметров и фазовых переменных, а также проверка возможности такой стыковки осуществляется во время сборки модели.

Общий вид оператора коммутации таков:

ОператорКоммутации :: = {ОпределительДиапазона} АдресПараметра "=" АдресФазовойПеременной ";".

Один оператор определяет один фрагмент связи, причем четко фиксируется направление связи - от контакта, указанного за равенством, к контакту, указанному перед ним. Синтаксис адресов контактов в операторах коммутации определяется формулами вида:

АдресПараметра, :: = {АдресКомпоненты.}ИмяПараметра
{ИндексПараметра}

АдресКомпоненты, :: = ИмяКомпоненты " (" ИндексКомпоненты ")"

АдресФазовойПеременной, :: = {АдресКомпоненты"." }ИмяФазовойПеременной
{ИндексФазовойПеременной}

Если адрес контакта сводится к его индексу, то это значит, что контакт принадлежит внешнему разъему описываемой компоненты модели. ИмяПодгруппы, ИмяОбъекта и ИмяПрибора - суть идентификаторы соответствующих классов компонент. Служебное слово INT помечает внутренние разъемы приборов. Термины ИндексПараметра, ИндексФазовойПеременной формально расшифровываются одинаково:

ИндексПараметра,
ИндексФазовойПеременной **::=** "(" ПростоеЦелоеВыражение ")".
ПростоеЦелоеВыражение **::=** ЦелоеБезЗнака |
[Целое Знак] "i*" ЦелоеБезЗнака |[Целое Знак] ЦелоеБезЗнака "*i" |[Знак] "i*"
ЦелоеБезЗнака Знак ЦелоеБезЗнака |[Знак] ЦелоеБезЗнака "*i" Знак ЦелоеБезЗнака.
Знак ::= "+" | "-".

Во второй формуле все варианты, кроме первого, относятся к случаю, когда записи оператора коммутации предшествует

Определитель диапазона **::=** "i=" Целое ".." Целое ":".

Это - конструкция, позволяющая одной записью оператора коммутации определить целую группу фрагментов каналов связи, а каких именно - ясно из контекста. Осталось только подчеркнуть, что указываемые в квадратных скобках номера подгрупп и объектов или приборов являются порядковыми во внутренней индексации той группы или объекта, в чей описатель войдет оператор. Примеры операторов коммутации:

```
i = 1..10 : ОбъектПервогоТипа (i).МассивПараметров(2*i-1) =
            ОбъектВторогоТипа (10-i).МассивФазовыхПеременных(2*i);
A1=Спутник.X;
Объект(0).x=x;
```

3.3. Синтаксис описателей комплексов

Двух предыдущих подразделов достаточно, чтобы без подробных дополнительных разъяснений определить принятые правила формирования спецификаций комплексов и компонент. Формальный синтаксис спецификации комплексов таков:

СпецификацияГруппы **::= COMPLEX** ИмяКомплекса**";"**
[ПараграфКомпонент]
[ПараграфФазовыхПеременных]
[ПараграфКонстант]
[ПараграфПараметров]

[ПараграфКоммутации].

ПараграфКомпонент ::= **COMPONENTS**
ПереченьСоставляющих; { ПереченьСоставляющих }
{END";"}

ПараграфФазовыхПеременных **::= PHASE**
ОпрСтруктДанных {**END ";"}**

ПараграфПараметров **::= PARAMETERS**
ОпрСтруктДанных {**END ";"}**

ПараграфКонстант **::= CONST**
ОпрСтруктДанных {**END ";"}.**

ПараграфКоммутации **::= COMMUTATION**
ГруппаКоммутации { ГруппаКоммутации. } **{END ";"}.**

Во всех параграфах, кроме последнего в комплексе, **END'ы** необязательны. Заголовок следующего параграфа автоматически означает конец предыдущего.

Смысл использованного здесь термина ОпрСтруктДанных разобран в предыдущем разделе, а несложные формулы определений других новых терминов выглядят следующим образом:

ПереченьСоставляющих **::=** ИмяКомпоненты " (" ЧислоЭкземпляров ")"
{"," ИмяКомпоненты "(" ЧислоЭкземпляров ")" } ";".
ГруппаКоммутации ::= [ОпределительДиапазона]
ОператорКоммутации.

В этих формулах ИмяКомплекса и ИмяКомпоненты суть имена классов компонент соответствующего типа, а формально – идентификаторы длиной не более 20 символов; ЧислоЭкземпляров – положительное целое, равное числу составляющих указанного перед скобкой класса; ОпределительДиапазона, ОператорыКоммутации - термины из предыдущего раздела.

3.3. Синтаксис описателей методов

Методами в излагаемой концепции являются или функции $\vec{f}_m(\vec{x},\vec{a},\Delta t), m = 1,..., M$, о которых шла речь в разделе 1.2, или функции $\mathbf{\Phi}_i(\vec{x},\vec{a})$, введенные в этом же разделе, с помощью которых определяется наступление событий. Это те части модели, которые могут вызываться как локально, так и распределенно. Считается, что в качестве параметров методу передается некая запись, тип которой описан, и она же возвращается методом. Так как тип этой записи, в особенности для удаленных методов, определяется разработчиком метода, а не разработчиком модели, возникает вопрос о коммутации полей параметров метода с полями фазовых переменных и/или параметров компоненты. Кроме того, методы, реализующие элементы процессов, разделяются по отношению с модельному времени на:

- сосредоточенные – происходящие мгновенно,
- распределенные – занимающие не менее одного модельного такта и дающие определенный результат своего выполнения в виде изменений внутренних переменных модели в конце каждого такта,
- условно-распределенные – занимающие не менее одного модельного такта, но дающие результат своего выполнения в виде изменений внутренних переменных модели лишь при полном завершении выполнения.

ОписательМетода ::= METHOD ИмяМетода **":"** ТипМетода **";"**

[ПараграфПараметровМетода]

ТипМетода ::= **FAST** | **CONV** | **SLOW | EVENT.**

По умолчанию тип метода, реализующего элемент процесса, считается **SLOW,** и в этом случае явное его указание разрешается опустить.

Синтаксис параграфа параметров метода такой же, как у параграфа фазовых переменных описателя комплекса. Поскольку это единственный параграф описателя, он обязан закончиться ключевым словом **END.**

3.3. Синтаксис описателей компонент

Формула спецификации компоненты такова:

ОписательКомпоненты :: = **COMPONENT** ИмяКомпоненты **";"**
[ПарграфФазовыхПеременных]
[ПараграфПараметров]
[ПараграфКонстант]
[ПараграфМетодов]
[ПараграфКоммутации]
[ПараграфСобытий]
[ПарграфПереключателей].

Как и в случае комплексов, **END** обязателен лишь в самом последнем параграфе. В параграфе методов спецификации компоненты одним или несколькими процессами перечисляются все выполняемые в данном процессе методы, начиная с корневого, с которого по умолчанию будет начинаться модельная жизнь каждого процесса. Имена методов должны быть уникальными в пределах спецификации: нельзя одинаково назвать два разнотипных метода одной компоненты, но использование одного имени в спецификациях разных компонент не возбраняется. Формально синтаксис параграфа методов определяется так:

ПараграфМетодов **::** = ПотокМетодов |
| ПараграфМетодов ПотокМетодов **{END ";"}.**
ПотокМетодов **::=** ИмяМетода { "," ИмяМетода } ";".

Здесь ИмяМетода - идентификатор длиной не более 20 символов.

Параграф коммутации синтаксически не отличается от параграфа коммутации компонент комплекса. Содержательный же его смысл в том, что параметры методов достаются разработчику модели от разработчиков методов такими, какими они были удобны последним, в частности, множество параметров метода может быть существенно уже множества фазовых переменных, параметров и констант компоненты, а может и совпадать или даже оказаться шире последнего. В любом случае, параграф призван указать, каким параметрам метода соответствуют те или иные фазовые переменные, параметры и константы компоненты.

ПараграфКоммутации **::= COMMUTATION**
ГруппаКоммутации { ГруппаКоммутации. } **{END ";"}.**
ГруппаКоммутации ::= [ОпределительДиапазона]
ОператорКоммутации.

Параграф событий определяет набор событий, связанных с данной компонентой. Определяется идентификатор события и условие его наступления.

ПараграфСобытий **::** = ОписаниеСобытия { ";" ОписаниеСобытия } ";" **{END ";"}**.
ОписаниеСобытия **::** = ИмяСобытия ":" МетодВычисляющийСобытие ";".

Параграф переключателей определяет автоматную функцию процесса. В нем для каждого элемента каждого процесса должны быть перечислены все разрешенные переходы, т.е. названы все

элементы, на которые прибор может переключаться по завершении данного, и указаны события, в зависимости от которых реализуется то или иное переключение. Формат параграфа таков:

ПарграфПереключателей **::= SWITCHES** АвтоматнаяФункция **{END ";"}**

АвтоматнаяФункция **::=** ПереключенияЭлемента
{ ПереключенияЭлемента }.

ПереключенияЭлемента :: = ИмяЭлемента ":" СписокПереходов.

СписокПереходов **::=** { ИмяЭлемента "," ИмяСобытия "."}ИмяЭлемента ";".

Понятно, что перед двоеточием во второй формуле стоит имя того элемента, переходы с которого определяет следующая за двоеточием конструкция. В последней перечисляются возможные переключения. Они упорядочиваются по важности событий, причем завершается перечень именем элемента, при котором не стоит никакого события. Это значит, что если не реализуется ни одно из вошедших в СписокПереходов событий, то произойдет переключение на элемент с указанным именем.

3.4. Компиляция описателей компонент модели

Описатели типов данных, компонент, методов компилируется в следующую систему связанных межу собою таблиц реляционной базы данных:

1. Типы_модели,
2. Поля_типов,
3. Компоненты_модели,
4. Процессы_модели,
5. Методы_модели,
6. Коммутация_методов,
7. Переключатели_модели.

Подробнее компиляция описателей и вид результирующих таблиц базы данных приведен в [4]. При компиляции описателя комплекса происходит его преобразование в «комплекс как компоненту» путем объединения констант, фазовых переменных, процессов, элементов, событий и, наконец, параметров. При этом из объединенных параметров исключаются те, что участвуют в коммутации компонент. При исключении параметра нужно проследить, с какими методами он коммутировал, и заменить в строках таблицы коммутации методов комплекса как компоненты поля, связанные с этим параметром, на поля соответствующей фазовой переменной.

Выбрать параметры, подлежащие исключению, можно с помощью запроса:

SELECT * FROM [Коммутация_компонент] WHERE [Номер комплекса] = n

Выбрать методы, подлежащие перекоммутации, можно по запросу:

SELECT [Номер метода] FROM [Коммутация_методов] WHERE [ФПК] = “параметры” AND [Поле компоненты] = ИмяПоля.

Наконец, поля фазовых переменных, на которые нужно заменить поля параметров в таблице «Коммутация_методов», также находятся в таблице первого из запросов.

4. Компоновка модели

4.1. Генерация базы данных модели

На стадии компиляции описателей модели формируется, как это было показано выше, база данных, описывающая состав компонент модели и связи между ними. Задача этапа компоновки – проверить полноту и непротиворечивость системы этих описаний и сгенерировать на их основе рабочую базу данных для последующего проведения имитационных экспериментов.

Исходным материалом для компоновки модели является результат компиляции компоненты или «комплекса как компоненты», а именно, таблицы, приведенные в предыдущем разделе.

На основе первой и четвертой из них строятся две новые таблицы: по имеющимся описаниям типов переменных выделяется реальное пространство или для всего имитационного эксперимента, или же для нулевого момента времени, с последующим выделением пространства для очередного шага имитации. Заполнение базы данных начальными условиями осуществляется вручную или же с помощью специальной программы, в данной работе мы не останавливаемся на технике заполнения базы данных модели начальными значениями, хотя этот процесс и может оказаться нетривиальным. Также таблица «Процессы модели» разворачивается в модельном времени, и на каждом шаге моделирования в ней будет указываться текущий элемент процесса.

4.2. Работа модели во время эксперимента

Предположим, что мы находимся в начале такта имитации. Задан некий стандартный шаг времени Δt.

Просматривая таблицу процессов, выполняем текущие сосредоточенные элементы, передавая им параметры в соответствии с таблицей коммутации методов. Возвращенные значения параметров присваиваем соответствующим фазовым переменным. В соответствии с таблицей переключателей, вычисляем следующие элементы.

- Повторяем предыдущий шаг, пока имеются сосредоточенные элементы.
- Выполняем со стандартным шагом Δt распределенные и условно-распределенные элементы. Вычисляем наступление событий, связанных с очередным элементом. В частности, такими событиями могут быть и заранее запланированные с помощью специальных системных средств, как в [1], моменты окончаний элементов. Как указывалось в [3], наступление события есть обращение в ноль соответствующей функции. Методы событий реализуют именно такие функции. Смена функцией события знака на концах отрезка Δt говорит о наступлении события внутри этого отрезка. Далее, с помощью того или иного метода аппроксимации, с приемлемой точностью находится момент наступления события. Минимум из этих времен по всем элементам даст продолжительность следующего шага имитации, после которого для закончившихся элементов в соответствии с таблицей переключателей и наступившими событиями вычисляются последующие.
- Модельное время переводится на вычисленную продолжительность прошедшего шага имитации.

Мы оказываемся в начале нового такта имитации, который снова пытаемся пройти исходя из стандартного шага Δt.

Заключение

По мнению авторов предложенная концепция могла бы применяться и более широко: не только как средство объединения в единый моделирующий комплекс различных имитационных моделей, описывающих слабо связанные части единого процесса, но и как универсальное средство совместного использования в сети разнородных и разноплатформенных информационных и алгоритмических ресурсов, что могло бы существенно расширить круг ее потенциального применения. Тем самым у инструментальных систем имитационного моделирования появился бы шанс превратиться из экзотических программных продуктов предназначенных для узкого круга специалистов в области имитации, в массовое и универсальное средство объединения ресурсов Интернета для совместного выполнения широкого круга задач.

В настоящее время выполняется разработка макета инструментальной системы, концепция которой описана в данной статье. Планируется, что макет будет состоять из системы программирования, базирующийся на языке описания комплексов и компонент. Эта система программирования должна включать редактор с графическим интерфейсом описания моделей и их компонент, компилятор, генерирующий, в частности базу данных модели, средства отладки. Испытания проектных решений, касающихся разработки инструментальной системы, выполняются с помощью мо-

дели, имитирующей экономические, демографические, экологические процессы в системе из нескольких стран. Эта модель описана в [1], а ее компьютерная реализация доступна по адресу http://simul.ccas.ru/.

Литература

1. Павловский Ю.Н., Белотелов Н.В., Бродский Ю.И. Имитационное моделирование. М.: «Академия», 2008, 236с.
2. Бродский Ю.И., Лебедев В.Ю. Инструментальная система имитации MISS. М.: ВЦ АН СССР, 1991, 180с.
3. Бродский Ю.И. К разработке концепции построения инструментальной системы распределенного моделирования //Моделирование, декомпозиция и оптимизация сложных динамических процессов, М.: ВЦ РАН, 2007, С. 14-34.
4. Бродский Ю.И. Описание, компоновка и работа модели в инструментальной системе распределенного моделирования //Моделирование, декомпозиция и оптимизация сложных динамических процессов, М.: ВЦ РАН, 2008, С. 24-46.

Бродский Юрий Игоревич. Ведущий научный сотрудник Вычислительного центра им. А.А. Дородницына РАН. Окончил Московский физико-технический институт в 1976 году. Кандидат физико-математических наук, автор более 70 научных работ, в том числе 5 монографий и 4 учебных пособий. Область научных интересов: математическое моделирование, имитационное моделирование сложных систем, распределенное моделирование, инструментальные системы имитационного моделирования. E-mail: brodsky@ccas.ru

Павловский Юрий Николаевич. Главный научный сотрудник Вычислительного центра им. А.А. Дородницына РАН. Окончил Московский физико-технический институт в 1960 году. Доктор физико-математических наук, профессор, член-корреспондент РАН, лауреат премии Совета Министров (1981), лауреат премии им. Винера (1992). Автор более 130 научных работ, в том числе более 10 монографий. Область научных интересов: групповой анализ, геометрическая теория декомпозиции, математическое моделирование, имитационное моделирование сложных систем, теория управления. E-mail: j_pvlsk@redline.ru.