УДК 681.32

СПОСОБЫ СОЗДАНИЯ УНИВЕРСАЛЬНОГО ИНСТРУМЕНТАРИЯ ДЛЯ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

Л. А. Панкова, В. А. Пронина

Институт проблем управления РАН им. В. А. Трапезникова, г. Москва

Рассмотрены современные технологии компьютерного моделирования и классификация компьютерных моделей. Выделены подходы к созданию универсального инструментария для компьютерного моделирования на основе современных технологий.

ВВЕДЕНИЕ

Компьютерное моделирование — мощное аналитическое средство, вобравшее в себя весь арсенал новейших информационных технологий, включая развитые графические оболочки для конструирования моделей и интерпретации выходных результатов моделирования, мультимедийные средства, поддерживающие анимацию в реальном масштабе времени, объектно-ориентированное программирование, Интернет-решения и др.

В настоящей работе описывается существующая классификация компьютерных моделей по степени дискретности переменных модели и по способу описания функционирования модели (через события, действия, процессы, агентов). Рассматриваются подходы к созданию универсального инструментария для компьютерного моделирования на основе современных технологий.

1. ТЕХНОЛОГИИ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

Начиная с середины 1960-х гг. широкое распространение получила структурно-функциональная [1—3] методология анализа, проектирования и разработки моделей — SADT (Structured Analyses and Design Technique — технология структурного анализа и проектирования) [4]. В основе SADT лежит принцип иерархической декомпозиции сверху вниз (системы AUTOIDEF0, Design/IDEF). Особенность моделирования на основе SADT состоит в возможности одновременно со структурированием проблемы разрабатывать структуру базы данных. Применение методологии SADT позволяет унифицировать различные блоки модели сложной системы, распараллелить процесс составления модели и объединить отдельные модули в единую иерархическую динамическую модель. Еще одним широко известным инструментальным средством структурно-функционального моделирования, основанным на SADT, является пакет Bpwin (Macro-Project). Он предназначен для моделирования и оптимизации бизнес-процессов. Его достоинство заключается в возможности связи с известным инструментальным средством разработки баз данных Erwin (Logic Works). Отечественное инструментальное средство для структурно-функционального анализа — CASE-Аналитик.

Однако структурно-функциональная технология моделирования характеризуется слабой взаимосвязью процессов и данных, присутствующих в модели. Многочисленные наблюдения за жизнедеятельностью систем любого рода позволяют сделать вывод, что данные модели на порядок более стабильны, чем процессы, которые происходят в ней. В отличие от структурных методологий, ориентированных на функциональность системы, объектные технологии [5], ориентированные на данные, на тесную взаимосвязь данных и процессов системы, позволяют программным системам быть более надежными, легко реализуемыми и устойчивыми к изменениям. Кроме того, такая система наиболее соответствует общим концепциям поведения систем реального мира.

Объектно-ориентированный подход в последнее время стал так прочно ассоциироваться с программированием, что многие забывают о его прямой связи с моделированием: первоначально он был применен в языке моделирования SIMULA-67 [6]. В дальнейшем объектно-ориентированный подход развивался почти исключительно программистами. И только в последние годы объектно-ориентированный подход стал востребованным в своей «родной области» — моделировании. Этот подход поддерживают современные системы моделирования (Omola, Dymola, Model Vision, AnyLogic). Кандидатом на роль объекта в объектно-ориентированном моделировании (ООМ) выступает компонент (блок) — совокупность переменных и поведения, взаимодейст-

вующий с внешним миром через внешние переменные. Компонент является экземпляром некоторого класса.

В итоге тридцатилетнего развития объектно-ориентированного подхода в первой половине 1990-х гг. был предложен разработанный под эгидой «Rational Software Corporation» на основе наиболее популярных объектных методов — Object Modeling Technique (OMT, James Rambaugh), Booch (Grady Booch) и Object Oriented Software Engineering (OOSE, Ivar Jacobson) — унифицированный язык объектного моделирования (Unified Modeling Language — UML) [5], предназначенный для создания объектно-ориентированных спецификаций моделей систем.

Уже сегодня существуют множество CASE-средств, автоматизирующих процесс анализа и проектирования в UML: Rational Rose, Paradigm Plus, Select Enterprise, Microsoft Visual Modeler for Visual Basic и др. Современные CASE-средства поддерживают множество языков программирования (C++, Java, Delphi, Power Builder, Visual Basic, Centura, Forte, Ada, Smalltalk и др.), осуществляя автоматическую кодогенерацию и обратное проектирование, т. е. построение модели по программному коду, а также позволяют генерировать базу данных для большинства из существующих SQL-серверов.

В 1999 г. разработан профиль UML для моделирования систем реального времени под названием UML-RT, который позволяет иерархически строить системы из инкапсулированных модулей с четко определенными интерфейсами (портами) и явными каналами связи (соединителями).

Структурная и объектная методологии моделирования сформировались значительно позже, чем появилась возможность имитировать реальность с помощью компьютерных моделей. В русском языке появились термины «имитационная модель» и «имитационная моделирование», а в английском — «simulation modeling». Если в английском языке термин имеет вполне четкий смысл, ибо симуляция и моделирование не являются синонимами, то по-русски имитационная модель — это нонсенс. Любая модель, в принципе, имитационная, ибо она имитирует реальность [4]. Указывая, что данная модель имитационная, подчеркивается, что в отличие от других типов абстрактных моделей в этой модели сохранены и легко узнаваемы такие черты моделируемого объекта, как структура, связи между компонентами, способ передачи информации. Традиционно под компьютерным моделированием понимается лишь имитационное моделирование. В дальнейшем термин «имитационное» будем опускать.

Инструментальные средства компьютерного моделирования, или языки моделирования, появились довольно давно, почти одновременно с языками Algol и Fortran, и прошли путь от бурного развития в 1970-х гг., когда они ежегодно рождались десятками, до современного стабильного состояния, когда доминирует лишь несколько языков [7].

Все существующие средства моделирования представляют собой оболочку над каким-либо языком программирования (Fortran, Algol, C, Pascal, Java и др.), ориентированную на определенный тип модели или на конкретную область применения. Тип модели определяется такими критериями, как способ отсчета времени, степень дискретности зависимых от времени переменных, задание способа функционирования модели, степень детерминированности модели, степень изолированности модели (активности проводимого эксперимента). Кроме того, специализация системы моделирования может быть связана с проблемной ориентацией (например, банковская система, угольная промышленность, автомобилестроение и т. д.).

Специализацией систем моделирования достигается максимальное упрощение описания задач и подготовки исходных данных для пользователя, не являющегося специалистом в области программирования и численного моделирования. Чем уже проблемная ориентация, тем меньше работы по программированию. В пределе можно добиться полной параметризации системы, и тогда задание конкретной модели будет состоять лишь в указании числовых значений параметров.

В современных системах моделирования компьютерная модель создается с помощью графического языка, потом автоматически транслируется на язык моделирования и далее — на машинный язык. Трансляция на машинный язык производится в два этапа: сначала — трансляция с языка моделирования на «родной» язык программирования, затем — трансляция с языка программирования на машинный компилятором соответствующего языка программирования.

2. КЛАССИФИКАЦИЯ КОМПЬЮТЕРНЫХ МОДЕЛЕЙ

В компьютерной модели время является основной независимой величиной. Другие переменные модели зависят от времени. Модельное время может быть непрерывным или дискретным в зависимости от того, могут ли дискретные изменения зависимых переменных происходить в любые моменты времени или только в определенные [8]. Традиционно модели делятся на дискретные, непрерывные и комбинированные в зависимости от поведения зависимых переменных. В дискретной модели зависимые переменные изменяются дискретно в определенные моменты модельного времени — моменты свершения событий. В непрерывной модели зависимые переменные изменяются непрерывно в течение модельного времени. В комбинированной (гетерогенной) модели зависимые переменные изменяются дискретно, непрерывно или непрерывно на отрезках времени.

Функционирование дискретной модели (Arena, Extend, SimProcess, AutoMod, PROMODEL, Enterprise Dynamics, FlexSim, eMPlant, GPSS, SIMULA, Q-GERT, SLAM, GASP, Witness, Taylor, QUEST, SIMFACTORY II.5, SIMPLE++, ARIS и др.) задается:

— определением изменений состояния модели (совокупности значений переменных), происходящих в моменты свершения событий;

— описанием действий, в которых принимают участие элементы модели;

— описанием процессов, через которые проходят элементы модели.

Событие определяет начало или окончание действия. Процесс — это последовательность событий, которая может состоять из нескольких действий [8].

Способ описания функционирования дискретной модели определяет три альтернативные методологические подхода к построению дискретной модели: событийный, сканирования активностей и процессно-ориентированный. Каждый подход предлагает некоторую схему описания модели и каждый обладает определенны-

ми достоинствами и недостатками. Процессно-ориентированный (SIMULA, SLAM, SIMAN) подход краток и прост в изучении схемы модели, но не обладает достаточной гибкостью. Событийный подход (SIMSCRIPT) — более сложен, но обеспечивает гибкую схему модели. Подход сканирования активностей (CSL) очень эффективен, когда продолжительность действия определяется в зависимости от того, насколько состояние модели удовлетворяет заданным условиям. Но из-за необходимости сканировать условия для каждого действия он менее эффективен по сравнению с событийным подходом.

В непрерывной модели состояние системы определяется непрерывно изменяющимися зависимыми переменными (переменными состояния). Непрерывная модель задается уравнениями для этих переменных (GSSL, Dynamo [9], Экспресс-радиус [10] и др.).

В комбинированных моделях различают два вида событий: временные события, которые совершаются в определенные моменты времени, и события состояния, которые происходят, когда модель достигает определенного состояния.

Частным случаем комбинированных (или расширением непрерывных) моделей являются так называемые гибридные модели — непрерывные модели, имеющие различное поведение в различных областях фазового пространства (Matlab-Simulink-Stateflow, Model Vision System, Modelica). Их фазовая траектория в зависимости от происходящих событий состояния, приводящих к смене поведения, оказывается то в одной области, то в другой. Таким образом, к гибридным можно отнести классические непрерывные модели, чье фазовое пространство разбивается на области с различным поведением, модели с разрывными правыми частями и модели, у переменных которых меняется размерность в различных областях фазового пространства. Каждой области можно поставить в соответствие вершину некоторого графа, а его направленные дуги трактовать как возможные пути смены текущего локального поведения. Границы областей обычно задают с помощью предикатов, которые приписываются соответствующим дугам графа. Таким образом, гибридная система может быть представлена в виде графа, вершинам которого поставлены в соответствие классические непрерывные модели и одна из вершин помечена как начальная, а дугам — условия смены поведения и мгновенные действия, выполняемые при смене поведения. Такая формализация называется гибридным автоматом. Это наиболее наглядная и удобная форма описания поведения гибридных моделей, являющаяся расширением карты состояний Харела, принятой в UML. По мере движения модельного времени фазовая траектория пересекает границы областей и меняется вид решаемых уравнений.

В работе [11] введены два типа гибридных автоматов:

— обобщенная карта состояний, применяемая для описания гибридных систем в Matlab-Simulink-Stateflow;

— карта поведения, применяемая для описания гибридных систем в Model Vision (в частном случае чисто дискретной модели всем узлам карты поведения следует приписать пустые локальные поведения и тогда карта поведения превращается в карту состояний Харела).

Традиционные модели (дискретные, непрерывные и комбинированные) характеризуются тем, что в них централизованно определяется поведение (динамика) модели в целом (моделирование сверху вниз). Разработка модели невозможна без знания о глобальных зависимостях исследуемого объекта.

Современный альтернативный подход к разработке модели называется агентным моделированием (AnyLogic [12], ЭКОМОД [13]). Общепризнанного определения «агента» не существует; до сих пор спорят о том, какими качествами должен обладать объект, чтобы «заслужить» называться агентом — инициативностью и реактивностью, ориентацией в пространстве, способностью обучаться, общаться, «интеллектом» и т. д. [12]. Определяется поведение агента на индивидуальном уровне, а глобальное поведение модели возникает как результат деятельности многих агентов, каждый из которых следует своим собственным правилам, живет в общей среде и взаимодействует со средой и с другими агентами. Поэтому агентное моделирование называют еще моделированием снизу вверх. Важное преимущество агентного моделирования в том, что разработка модели возможна в отсутствии знания о глобальных зависимостях. Агентная модель строится на основании локальных поведений участников процесса, которые и определяют глобальное поведение системы. Кроме того, агентную модель проще поддерживать: уточнения обычно делаются на локальном уровне и не требуют глобальных изменений.

В работе [12] показано, как построить агентную модель по существующей непрерывной или дискретной, и демонстрируется, как она может быть расширена для учета более сложных поведений, зависимостей и взаимодействий.

3. ПОДХОДЫ К СОЗДАНИЮ УНИВЕРСАЛЬНОГО ИНСТРУМЕНТАРИЯ ДЛЯ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

Подход первый к созданию универсального инструментария для компьютерного моделирования основан на «объединении» существующих систем моделирования [14]. В качестве компонентов универсального пакета предлагается взять уже готовые компоненты существующих пакетов, согласовав интерфейс этих компонентов.

«Идеальный» универсальный пакет моделирования должен:

— поддерживать типовые модели;

— иметь для каждой типовой модели входной язык;

— создаваться на принципах ООМ;

— представлять открытый программный продукт из типовых модулей, прототипы которых уже существуют.

Подход второй к созданию универсального инструментария для компьютерного моделирования основан на применении объектного языка программирования и современных технологий модуляризации (COM, CORBA) [15].

Модульная структура моделирующих программ практически неизменна:

— графический интерфейс — представление математической модели в виде, понятном пользователю (структурные схемы, схемы физические принципиальные, гибридные карты состояний и пр.);

— система управления базой данных (СУБД) — хранение объектов модели, а также требуемые преобразования структуры СУБД;

— математическое ядро — исполнение потоков математических функций в процессе функционирования модели;

— серверы визуализации и online-воздействий — интерфейс между функционирующим математическим ядром и пользователем.

Все перечисленные модули уже традиционно считаются независимыми программными продуктами. Разработчики моделирующих программ при создании своих продуктов, как считают сторонники этого подхода, должны придерживаться современных технологий модуляризации (COM, CORBA), а не делать все самостоятельно.

В работе [15] представлен обзор программных решений на языке C++, которые могут быть положены в основу математических ядер моделирующих программ с поточной моделью управления и могут быть выполнены в виде COM-серверов. Показана возможность интеграции математических ядер разных производителей с редактором векторной графики Visio. Сформулированы задачи модулей стыковки основных компонентов программ моделирования.

Подход третий основан на расширении унифицированного языка объектного моделирования UML.

В работе [11] для построения универсальной системы моделирования предполагается создать язык современного объектного моделирования (OOML) путем «скрещивания» языка UML и современных достижений в области непрерывного моделирования (в том числе, идей, предложенных в языке Modelica [16]). Понятия объекта, инкапсуляции, наследования, полиформизма применительно к моделированию имеют ряд важных особенностей. В языке «физического моделирования» Modelica в настоящее время наиболее широко используются идеи объектного программирования. Однако специалисты считают, что он не может быть положен в основу гипотетического языка OOML, так как недостаточно универсален: решает проблемы компонентного моделирования с неориентированными блоками, а средства описания дискретных аспектов поведения «неудобны». Рассматривается расширение языка UML, учитывающее особенности моделирования гибридных систем.

С другой стороны, базовые механизмы UML (стереотипы, маркированные значения и ограничения) на основе существующих элементов уже позволяют создавать расширения (так называемые профили), ориентированные на определенную предметную область. Так, UML-профиль исполнения — UML Profile for Schedulability, Performance and Time Specification [17] — позволяет аннотированные UML-диаграммы проектируемого приложения использовать для создания дискретно-событийных моделей проекта на ранней стадии [18]. А разработанный гибридный UML-профиль, основанный на языке CHARON [19], может служить для построения гибридных моделей [20].

ЗАКЛЮЧЕНИЕ

В работе сделан обзор существующих средств компьютерного моделирования: описана устоявшаяся классификация моделей, традиционные и современные технологии создания средств моделирования. Выделены наметившиеся подходы к созданию универсального инструментария для компьютерного моделирования на основе современных технологий:

- «объединение» существующих систем моделирования;
- использование объектного языка программирования и современных технологий модуляризации COM, CORBA;
- расширение унифицированного языка объектного моделирования UML.

Очевидно, что последний из этих подходов в наибольшей степени соответствует уровню требований, предъявляемых к современным инструментам для разработки систем имитационного моделирования.

ЛИТЕРАТУРА

1. *Вендров А. М.* CASE-технологии. Современные методы и средства проектирования информационных систем. — М.: Финансы и статистика, 2003.
2. *Гэйн К., Сарсон Т.* Структурный системный анализ: средства и методы. — М.: Эйтекс, 1993.
3. *Калашян А. Н., Калянов Г. Н.* Структурные модели бизнеса: DFD-технологии. — М.: Финансы и статистика, 2003.
4. *Моисеев Н. Н.* Математические задачи системного анализа. — М.: Наука, 1981.
5. *Фаулер М., Скотт К.* UML в кратком изложении. Применение стандартного языка объектного моделирования. — М.: Мир, 1999.
6. *Дал У. И., Мюрхауг Б., Нюгорд К.* Универсальный язык моделирования. — М.: Мир, 1969.
7. *Бахвалов Л. А., Микулич Л. И.* Компьютерное моделирование: основные тенденции развития инструментальных средств // Труды ИПУ. — М., 1999. — Т. 2. — С. 5—21.
8. *Прицкер А.* Введение в имитационное моделирование и язык СЛАМ II. — М.: Мир, 1987. — 644 с.
9. *Дж. Форрестер.* Мировая динамика. — М.: Наука, 1978.
10. *Дорри М. Х., Рощин А. А.* Инструментальные средства «Экспресс-Радиус» для автоматизации динамических расчетов систем управления // Приборы и системы управления. — 1996. — № 3.
11. *Беньковић Е. С., Колесов Ю. Б., Сениченков Ю. Б.* Практическое моделирование сложных динамических систем. — СПб.: БХВ, 2002. — 441 с.
12. *Борщев А.* От системной динамики и традиционного ИМ — к практическим агентным моделям: причины, технология, инструменты. <http://www.xjtek.com/files/papers/fromsdtoabmru.pdf>.
13. *Экомод* — интеллектуальный инструмент разработки и исследования динамических моделей экономики / А. А. Петров, И. Г. Поспелов, Л. Я. Поспелова, М. А. Хохлов // Вторая всероссийская научно-практическая конф. по вопросам применения имитационного моделирования в промышленности «Имитационное моделирование. Теория и практика». — Москва, 2005.
14. *Колесов Ю. Б., Сениченков Ю. Б.* Компьютерное моделирование. <http://cnews.ru>.
15. *Клиначев Н. В.* Обзор архитектурного построения программ математического моделирования динамических систем. <http://www.vissim.nm.ru/simkernel.html>.
16. <http://www.modelica.org>.
17. *UML* profile for schedulability, performance and time specification. <http://www.omg.org/docs/ptc/03-03-02.pdf>.
18. *Balsamo S., Grosso M., and Marzolla M.* Towards simulation-based performance modeling of UML specifications. <http://www.dsi.unive.it/~perf/publications.php>.
19. *CHARON* toolkit and information. <http://www.cis.upenn.edu/mobies/charon>.
20. *HybridUML* Profile for UML 2.0 / Kirsten Berkenk, Stefan Bisanz, Ulrich Hannemann, Jan Peleska. <http://www-verimag.imag.fr/EVENTS/2003/SVERTS/PAPERS-WEB/23-Hannemann-hybridUMLRT.pdf>.

☎ *(495) 334-92-49;*

e-mail: pron@ipu.ru ❑