УДК: 681.3.06(075.8)

Ю.Б. Колесов, Ю.Б. Сениченков

# Объектно-Ориентированное Моделирование в среде Rand Model Designer

# Object-Oriented Modeling with Rand Model Designer

*Аннотация*. Среда визуального моделирования Rand Model Designer предназначена создания иерархических многокомпонентных событийно-управляемых моделей сложных динамических систем. Среда использует объектно-ориентированный язык моделирования, соответствующий UML-«стандарту», и позволяет создавать модели постоянной и переменной структуры из «ориентированных» и «неориентированных» блоков, с поведением, определяемым внутренними машинами состояний.

*Ключевые слова*. Объектно-ориентированное моделирование, сложные динамические системы, среды визуального моделирования.

*Abstract*. Rand Model Designer is a modern tool for modeling and simulation hierarchical multicomponent event-driven dynamical systems. It has UML-based object-oriented Model Vision Language for designing dynamical and hybrid systems using modification of State Machines, large-scale multicomponent systems: control systems with «inputs-outputs», «physical» systems with «contacts-flows», and (new!) variable structure component systems, particularly «agent» systems.

*Key words*. Object-oriented modeling, complex dynamical systems, visual tools for modeling and simulation.

Объектно-ориентированное моделирование (ООМ) – это современная технология компьютерного моделирования, широко использующаяся в

научных исследованиях, проектировании технических систем и анализе бизнес-процессов, и позволяющая создавать надежные компьютерные модели быстрее и дешевле.

Компьютерное моделирование появилось практически одновременно с появлением первых компьютеров. В те времена компьютерные модели создавались вручную с использованием языка программирования Фортран или даже языка Ассемблера. Например, одному из авторов довелось разрабатывать свою первую компьютерную модель, имитатор еще не разработанной аппаратуры на испытательном стенде, в машинных командах БЭСМ-4. Однако, уровень абстракции моделируемой системы обычно существенно выше уровня абстракции языка программирования. Соответствие между моделью и ее программной реализацией разработчику приходилось держать «в голове», что являлось источником многочисленных ошибок. Часто для воспроизведения поведения непрерывных систем использовались свои собственные программные реализации численных методов, что приводило к получению сомнительных результатов. Частично последняя проблема была решена с помощью профессиональных библиотек (1980-1983): LINPACK, EISPACK, LAPACK (вычислительные методы линейной алгебры), ODEPACK (численные методы решения обыкновенных дифференциальных уравнений).

Начиная с 50-х годов прошлого века начали появляться высокоуровневые языки моделирования и поддерживающие их программные инструменты, создававшие исполняемую компьютерную модель автоматически. Одним их первых языков моделирования был язык GPSS [10], предназначенный для моделирования систем массового обслуживания. Этот язык интересен тем, что в нем по существу использовался объектно-ориентированный подход, хотя такие слова явно не произносились. Транзакции на входе моделируемой системы создавались как экземпляры стандартного класса «Транзакция»,

атрибуты которых имели различные значения, и, которые уничтожались по окончании работы.

В 1967 году появился язык моделирования Simula-67 [11], в котором впервые явно использовались все знаковые для ОММ слова – класс, экземпляр, наследование, полиморфизм. Однако, в то время этот революционный проект не получил широкого распространения в практике моделирования. Это было связано прежде всего с ограниченными вычислительными возможностями тогдашних компьютеров, так как накладные расходы при использовании ООМ были велики. В те годы основное внимание уделялось однокомпонентным изолированным математическим моделям, записанным в форме больших систем уравнений, каждая из которых являлась уникальной. Проблемы структуризации и модификации моделей отступали на второй план, по сравнению с необходимостью повышать скорость и точность вычислений. Это и привело к тому, что полноценное применение объектно-ориентированного подхода в моделировании задержалось на десятилетия. Зато идеи, заложенные в языке Simula-67, способствовали появлению ряда объектно-ориентрованных языков программирования, некоторые из которых (C++, Object Pascal) успешно используются и сейчас.

В 1980 появилась, пожалуй, наиболее известная «универсальная» среда моделирования - пакет MATLAB. В пакете MATLAB предусмотрена возможность описывать изолированную однокомпонентную динамическую систему в векторно-матричной форме, вызывать встроенный решатель обыкновенных дифференциальных уравнений и другие решатели из профессиональных систематизированных коллекций, и визуализировать решение в форме временных и фазовых диаграмм. Несмотря на то, что модели в пакете MATLAB описывались не на математическом, а на алгоритмическом языке, похожем на Фортран, но более высокого уровня, это было большим шагом вперед и обеспечило пакету заслуженный успех.

Пакет MATLAB был ориентирован не только на создание традиционных математических моделей, но и имел специальную подсистему для компонентного моделирования. Эта подсистема в 1992 году получила современное название Simulink [4]. Графический язык пакета Simulink, привычный для разработчиков систем управления и использовавшийся еще в аналоговых машинах, позволяет собирать модель из типовых блоков, имеющих реальные физические аналоги в технических устройствах. Уравнения, с которыми не всегда дружны инженеры, отступают на второй план, а графический язык блок-схем становится главным средством описания моделей. Подсистема Simulink использует язык блок-схем для описания моделей и берет на себя трудоемкий для инженера процесс составления системы уравнений, соответствующей всей модели, передаваемой затем решателям пакета MATLAB, а результат их численного решения - различным «визуализаторам».

Технология визуального моделирования пакета Simulink сводится к сборке моделей из готовых «кубиков», элементов стандартных библиотек, с настраиваемыми параметрами. По существу здесь снова неявно проявляется технология ООМ: «кубики» - это экземпляры библиотечных классов. Удачно сконструированная библиотека базисных классов позволяет создавать все более сложные классы, используя готовые, проверенные конструкции в качестве элементов. Структурная сложность новых классов маскируется с помощью многоуровневой иерархической структуры и сложные классы на верхнем уровне выглядят как элементарные блоки. Однако, для дополнения или модификации набора базисных классов, необходимо использовать алгоритмический внутренний язык MATLAB.

Реальное проникновение ООМ в практику моделирования произошло только в конце 90-х годов прошлого века, и связано это с появлением «универсального языка моделирования» UML [1] и языка моделирования

«физических систем» Modelica [2] вместе с поддерживающими их инструментами моделирования.

Язык UML предназначен для разработки спецификаций сложных программно-аппаратных систем на ранних стадиях разработки. Его значение для дискретно-событийного моделирования состоит прежде всего в том, что он «канонизировал» сложившиеся в области программирования понятия объектно-ориентированного подхода и, став фактическим стандартом объектно-ориентированного подхода, заставил всех, кто его использует, говорить на одном языке. Кроме того, эти понятия были соединены с исключительно удобными и наглядными диаграммами (машинами) состояний, придуманными еще в 80-х годах прошлого века Д. Харелом. Язык UML легко расширяется для моделирования непрерывно-дискретных (гибридных) систем. Своеобразным индикатором популярности UML может служить тот факт, что в пакете Matlab появилась новая подсистема StateFlow, поддерживающая диаграммы состояний.

Язык Modelica решил проблему автоматизации компонентного моделирования при использовании компонентов с ненаправленными связями. Внешние переменные компонентов моделей Simulink называются «входами-выходами», что подчеркивает принципиальное значение направления передачи информации. Однако, во многих прикладных областях направленность внешних переменных отсутствует – например, направления токов и знаки напряжений в электрических схемах условны. Соединение внешних переменных различных блоков в электрических цепях означает всего лишь равенство напряжений и равенство нулю суммы токов. Это принципиально меняет методику построения итоговых уравнений всей системы и приводит к значительным проблемам. Авторы языка Modelica преодолели возникшие трудности. Использование компонентов с ненаправленными связями сразу же расширило область применения компонентного моделирования. Ранее такие

модели могли автоматически строиться только из ограниченного набора базовых компонентов, которые не могли быть созданы на входном языке моделирования. Теперь язык Modelica позволил средствами самого языка создавать библиотеки компонентов для электрических, гидравлических, механических и других «физических» систем. Пользователь может строить из этих компонентов любые сложные структуры, итоговые уравнения для которых будут построены автоматически аналогично «блочным» моделям Simulink. Классы Modelica могут наследоваться, доопределяться и переопределяться.

В настоящее время существует ряд инструментов моделирования, поддерживающих полноценную технологию ООМ и не ориентированных на какую-то узкую прикладную область. Это инструменты можно разделить на две группы:

1) инструменты, ориентированные на язык UML и, очень важную его часть, формализм машин состояния (гибридных автоматов);
2) инструменты, ориентированные на язык Modelica (механизмы, поддерживающие технологию «физического моделирования).

К первой группе относятся такие среды моделирования как Ptolemy (университет Беркли, США) [3], AnyLogic (The AnyLogic Company, Санкт-Петербург) [5] и Rand Model Designer (исследовательская группа MVSTUDIUM Group, СПбГПУ) [8,9]. Сюда же следует отнести «вечно молодую» среду Matlab, которая сейчас является сложно устроенной системой компонентов «MATLAB+Simulink+StateFlow+ToolBoxes». Признать среду Matlab объектно-ориентированной можно только с большими оговорками – полноценно технологию ООМ поддерживает лишь подсистема StateFlow. Однако это не мешает пакету Matlab занимать ведущее место в современном практическом моделировании.

Ко второй группе относятся среды, разработанные в рамках европейского проекта «Modelica», такие как Dymola, OpenModelica, MathModelica и другие.

Мы считаем среды, основанные на технологии ОММ, наиболее перспективными, и будем в дальнейшем говорить только о них.

Современная среда моделирования, претендующая на универсальность, должна позволять разрабатывать в рамках технологии ООМ следующие виды моделей:

- однокомпонентные непрерывные модели;
- однокомпонентные дискретно-событийные модели;
- однокомпонентные гибридные модели;
- многокомпонентные модели с непрерывными, дискретными или гибридными компонентами и ориентированными связями («блочные модели»);
- многокомпонентные модели с непрерывными, дискретными или гибридными компонентами и неориентированными связями («физические модели»);
- многокомпонентные модели с переменным составом компонентов и переменной структурой связей.

Инструменты первой группы, базирующиеся на UML, прекрасно поддерживают все эти виды моделей кроме «физических». Это связано с экспоненциальным ростом числа состояний в машине состояний, соответствующей всей модели. Действительно, если в модели всего лишь два компонента, и каждый имеет свою машину состояний только с двумя состояниями – совокупная машина состояний будет иметь уже четыре состояния, и каждому совокупному состоянию будет соответствовать своя совокупная система решаемых уравнений. Для компонентов с ненаправленными связями построение совокупной системы уравнений не сводится к простому механическому объединению компонентных уравнений, что характерно для компонентов с направленными связями, а требует в общем случае очень сложного анализа и преобразования уравнений. Пакет Matlab

позволяет работать с «физическими» моделями, но только в рамках базовых наборов компонентов.

Язык Modelica поддерживает «физическое моделирование», но использует свою концепцию объектов, не совпадающую с концепцией языка UML, а также специальные конструкции для описания дискретных событий в гибридных моделях. Достаточно сложный анализ совокупной системы уравнений проводится на стадии компиляции модели, но это оказывается возможным только для небольшого класса гибридных моделей: когда число искомых переменных и решаемых уравнений не меняется при переключениях.

Практика моделирования показывает, что машина состояний языка UML все же гораздо удобнее и нагляднее для описания дискретно-событийных и гибридных моделей, чем описание, предложенное авторами языка Modelica, и что существует много практически значимых гибридных моделей, с с которыми трудно справиться, используя язык Modelica. Кроме того, в рамках подхода Modelica достаточно сложно естественным образом моделировать системы с переменным составом.

В инструменте Rand Model Designer сделана попытка соединить сильные стороны обоих направлений: поддерживать «физическое моделирование» как это предложено в языке Modelica, и использовать при этом объектную парадигму и машину состояний языка UML. «Расплатой» за это решение явилась необходимость выполнения части анализа совокупной системы уравнений на стадии выполнения модели при каждом переключении. Оказалось, что этот анализ можно проводить с помощью алгоритмов «линейной сложности», и создаваемые с помощью RMD промышленные модели из компонентов с ненаправленными связями успешно работают в реальном времени. Отличия языка MVL, входного языка среды Rand Model Designer, от языка Modelica рассмотрено в [6,7].

Версия Rand Model Designer 7 позволяет создавать модели всех перечисленных видов на основе технологии ООМ. Пробная версия Rand Model Designer 7, доступна на сайте «www.mvstudium.com».

*Литература*


1. Буч Г., Якобсон А., Рамбо Дж ,UML 2.0,СПб.: Питер, 2006, 735с.
2. Modelica Association, Modelica® - An Unified Object-Oriented Language for Systems Modeling, Language Specification version 3.3. Modelica Association, 2012. - https://www.modelica.org (accessed in 2013).
3. Claudius Ptolemaeus, Editor, System Design, Modeling, and Simulation using Ptolemy II, Ptolemy.org, 2014. – http://ptolemy.org/books/Systems.
4. Дьяконов В.П. Simulink: Самоучитель. – М.: ДМК-Пресс, 2013. – 784с.
5. Ю.Карпов. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. – СПб.: БХВ-Петербург, 2005. – 400с.
6. Martin-Villalba C., Urquia A., Senichenkov Y., Kolesov Y. Two approaches to facilitate virtual lab implementation, Computing in Science and Engineering, Vol.16, No.1, January/February 2014, pp.79-86.
7. Ю. Б. Колесов, Ю. Б. Сениченков, А. Уркия, К. Мартин-Виллалба. Гибридные системы. Сравнительный анализ языков моделирования Modelica и Model Vision Language. Университетский научный журнал, №8 (2014), сс.102-111.
8. Колесов Ю.Б., Сениченков Ю.Б. Моделирование с систем. Динамические и гибридные системы. С. Петербург, БХВ, 2006., 224с.
9. Колесов Ю.Б., Сениченков Ю.Б. Математическое моделирование гибридных динамических систем: учеб. пособие. – СПб.: Изд-во Политехн. ун-та, 2014. – 236с.
10. Шрайбер Т. Дж. Моделирование на GPSS. — М.: Машиностроение, 1980. — 592 с.

11. Дал У.И., Мюрхауг Б., Нюгорд. К. Симула-67. Универсальный язык программирования. - М.: Мир, 1969. - 98с.

*Сведения об авторах*

Колесов Юрий Борисович
ЗАО «MB Софт»
Рабочий адрес:
Ученая степень, звание: д.т.н,
Должность: начальник отдела
Электронная почта: ybk2@mail.ru
SPIN-код: 0000-0000

Сениченков Юрий Борисович
Политехнический университет Петра Великого
Рабочий адрес: 195251, Санкт-Петербург, Политехническая ул., д.29
Ученая степень, звание: д.т.н., доцент
Должность: профессор
Электронная почта: petrov@inbox.ru
SPIN-код: 0000-0000

Kolesov Yuri Borsivotch
«MV Soft», Russia
Postal address:, 28-79, Rusakovskaya St., Moscow, 107014, Russia

Senichenkov Yuri Borsivotch
Peter the Great St. Petersburg Polytechnic University
Postal address: 29, Politechnicheskaya St., St. Petersburg, 195251, Russia