УДК 004.942

Исаков Андрей Алексеевич

ЭКСПЕРИМЕНТАЛЬНАЯ ОБОЛОЧКА OPENMVLSHELL

Аннотация

Статья посвящена программному продукту с открытым кодом OpenMVLShell (http://DCN.FTK.SPBSTU.RU), предназначенному для моделирования сложных динамических систем.OpenMVLShell опирается на объектно-ориентированный подход в моделировании и использует язык ModelVisionLanguage (MVL, семейство сред визуального моделирования MvStudium). Проект OpenMVL, в рамках которого разработана среда OpenMVLShell, рассматривается как первый шаг к разработке стандарта сред визуального моделирования сложных динамических систем. Открытость кода позволяет реализовывать новые решения и сравнивать с уже существующими. Продукт, как и его аналог OpenModelica, можно использовать в учебном процессе. Рассмотрены архитектура среды, приведены примеры использования.

Ключевые слова: среды моделирования, стандарты, структура пакетов моделирования, языки моделирования, объектно-ориентированный подход, численные методы.

ВВЕДЕНИЕ

В 1968 году был разработан стандарт **CSSL** (*Continuous System Simulation Language* – язык моделирования непрерывных систем), который стал вехой в развитии стандартов. Он обобщил идеи и языковые структуры существующих сред моделирования, описал минимальный набор возможностей таких приложений и определил структуры моделей. Стандарт CSSL предлагает структуры и возможности для *описания модели* и для *экспериментирования* с нею. Структура ядра системы иллюстрируется на рис. 1 вместе с элементами, обслуживающими дискретное поведение модели.

Математический базис ядра среды моделирования – это описание в пространстве состояний вида:

$$\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t), \vec{u}(t), t, \vec{p}), \quad \vec{x}(t_0) = \vec{x}_0, \quad (1)$$

где $\vec{x}(t)$ – переменные состояния, $\vec{u}(t)$ – переменные управления, $\vec{p}$ – параметры модели.

Это описание используется решателями ОДУ. Какое бы ни было исходное описание модели, оно должно быть приведено к (1). В этой схеме вектор производных $\vec{f}(\vec{x}, \vec{u}, t, \vec{p})$ используется, чтобы вычислить новое состояние $x_{i+1} = \Phi(\vec{x}_i, f_i, h)$, где h – размер шага (всё контролируется ядром моделирования), в дискретных моментах времени t_i.

Описанная форма задания модели делает стандарт CSSL не подходящим для современных технологий моделирования, потому как на практике в сегодняшних средствах моделирования общепринятыми являются описание моделей в форме АДУ, ОДУ, НАУ – значительно более широкое многообразие моделей.

Современные пакеты компьютерного моделирования давно отошли от стандарта CSSL, и каждый развивается своим путем, наращивая функционал, востребованный на практике. Данные практические возможности, никак не отраженные в стандарте CSSL, были выделены в статье [2]. Авторы в данной работе определяют распространенные и структурные возможности сред моделирования и сравнивают наиболее популярные современные пакеты компьютерного моделирования в отношении данных возможностей (см. табл. 1).

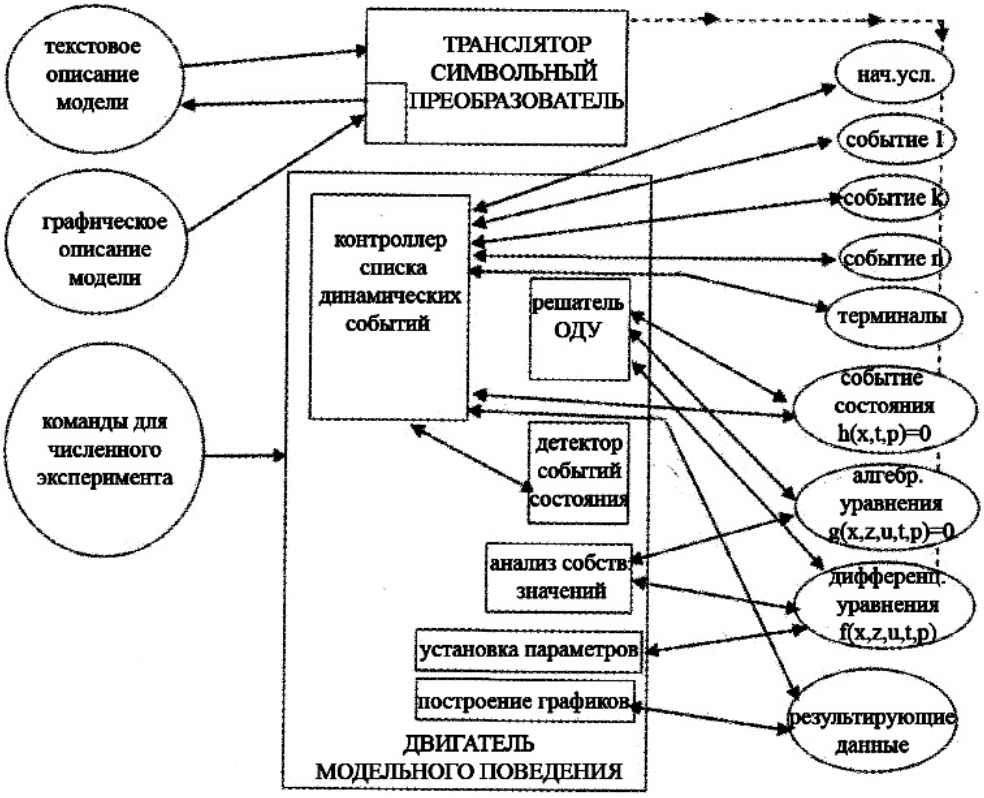

Рис. 1. Расширенная структура сред моделирования на базе стандарта CSSL с дискретными элементами

Сравнение возможностей, по мнению автора, – правильный шаг на пути создания стандарта в области создания пакетов компьютерного моделирования. Опираясь на потребности пользователей, можно выделять в средах визуального моделирования общее и понимать, что является спецификой конкретно взятого пакета моделирования.

ПРЕДПОСЫЛКИ СОЗДАНИЯ СТАНДАРТА НА РАЗРАБОТКУ

Проблема отсутствия общепринятого стандарта среди пакетов компьютерного моделирования ведет к тому, что для каждой среды разработки моделей реализуется своя библиотека стандартных моделей. Это всегда уникальные коллекции, иногда продаваемые отдельно от пакетов компьютерного моделирования, которые представляют отдельный интерес для обучения моделированию и исследования пакета на предмет его возможностей, производительности и точности воспроизведения решения. Конвертирование моделей из одной среды в другую часто оказывается трудоемким ручным процессом, трудно поддающимся формализации. Более того, даже аккуратно написав одну и ту же математическую модель в различных пакетах, сложно гарантировать (если модель не тривиальна), что в пакет будет загружена эквивалентная программная реализация данной модели. Ввиду того, что профессиональные пакеты компьютерного моделирования стоят достаточно дорого и исходный код таких разработок является закрытым, не представляется возможным посмотреть, с каким именно представлением работает пакет на этапе проведения численного эксперимента. Соответственно также нельзя посмотреть вид совокупной математической системы, которую пакет передаст численным методам для непосредственного решения, с какими настройками будет запущен тот или иной метод решения.

Табл. 1. Доступность распространенных и структурных возможностей в пакетах визуального моделировани

	Классификация моделей	Описание событий	Обработка событий состояния	Решение АДУ	Понижение индекса	Физическое моделирование (текст)	Физическое моделирование (граф.)	«Онлайн» визуализация	Карта поведения (текст)	Карта поведения (граф.)	Структурно-динамический анализ	Частотный анализ
MATLAB	нет	нет	(да)	(да)	нет	нет	нет	(да)	нет	нет	да	да
Simulink	да	(да)	(да)	(да)	нет	нет	(нет)	(да)	нет	нет	нет	да
MATLAB/Simulink	да	да	да	(да)	нет	нет	(нет)	(да)	нет	нет	да	да
Simulink/Stateflow	да	да	да	да	нет	нет	(нет)	(да)	(да)	да	нет	да
Simulink/Simscape	да	да	да	да	(да)	да	да	(да)	нет	нет	нет	да
Simulink/Simscape/Stateflow	да	да	да	да	(да)	да	да	(да)	(да)	да	нет	да
Matlab/SL/SS/Stateflow	да	да	да	да	(да)	да	да	(да)	(да)	да	да	да
ACSL	да	да	да	да	нет	нет	(нет)	(да)	нет	нет	нет	да
Dymola	да	да	да	да	да	да	да	да	(да)	(да)	нет	(нет)
MathModelica	да	да	да	да	да	да	да	(да)	(нет)	(да)	нет	(нет)
MathModelica/Mathematica	да	да	да	да	да	да	да	да	(нет)	(да)	да	да
Misilab	да	да	да	да	(нет)	да	да	(нет)	да	да	да	нет
OpenModelica	да	да	да	да	да	да	(нет)	(нет)	(нет)	(да)	нет	нет
SimulationX	да	да	да	да	да	да	да	да	(нет)	(да)	нет	да
AnyLogic	да	да	(да)	(да)	нет	нет	нет	да	да	да	да	нет
Model Vision Studium	да	да	да	да	да	да	да	да	да	да	да	да
Scilab	да	нет	(да)	(да)	нет	нет	нет	(да)	нет	нет	да	да
Scicos	да	(да)	да	да	(да)	да	да	(да)	да	(да)	нет	нет
Scilab/Scicos	да	да	да	да	(да)	да	да	(да)	да	(да)	да	да
MapleSim	да	(да)	(да)	да	да	да	да	да	нет	нет	(да)	(да)

Для того чтобы удовлетворить практически потребности пользователей, наиболее распространенные из которых приведены в табл. 1, разработчики реализуют в пакете компьютерного моделирования некоторые (иногда эвристические) алгоритмы решения научно-технических задач, возникающих в процессе работы. Закрытость исходного кода пакетов не позволяет сравнивать эти алгоритмы, говорить об эффективности различных реализаций. Обо всем можно судить только косвенно по общей производительности программного обеспечения.

Высокая стоимость пакетов моделирования в совокупности с закрытостью программного решения, принятого разработчиками среды визуального моделирования, с точки зрения обучения, не дают возможности познакомиться с внутренней структурой пакета, с теми проблемами, с которыми автоматически должен справляться инструмент моделирования. Специалист, понимающий внутреннюю специфику работы пакетов моделирования, будет являться на порядок более квалифицированным, нежели специалист, который только освоил правила входного языка описания моделей и научился менять настройки визуальной модели. Часто непонимание особенностей работы пакета, трактуется пользователями как ошибка, тогда как в действительности модель пользователя приводит к некорректной

математической системе, которую заведомо невозможно (или практически невозможно) решить.

Все данные аспекты являются предпосылками к созданию современного стандарта в области моделирования, который бы выполнял следующие функции:

1) классификация основных типы моделей (НАУ, ОДУ, АДУ);

2) унификация входного языка описания моделей (на базе UML);

3) декларирование стандартных инструментов работы с моделью, исследования ее поведения;

4) определение фундаментальных проблем, которые должны быть разрешены в пакетах компьютерного моделирования, чтобы обеспечить в них автоматическое выполнение всех этапов численного эксперимента и реализовать при этом поддержку важных практических функций (табл. 1):

– перевод данных проблем на язык математики и теории алгоритмов,

– предложение имеющихся в настоящей теории и практике решений сформулированных проблем;

5) определение структуры пакета, которая обладала бы свойство расширяемости;

6) предоставление библиотеки модельных задач и типовых принципов ее решения.

В силу того, что успешные профессиональные пакеты компьютерного моделирования постоянно наращивают список предоставляемых пользователю возможностей и на практике оказываются востребованы все новые возможности подобного программного обеспечения (которые уже не укладываются в табл. 1), декларируемый стандарт должен динамически развиваться, то есть:

1) содержать актуальный список возможностей пакетов визуального моделирования;

2) отражать наиболее современные придуманные разработчиками решения;

3) доопределять спецификацию входного языка, отражающую поддержку новых возможностей пакетов;

4) пополняться новыми модельными задачами, отражающими новые классические задачи, встречающиеся на практике.

Таким образом, для того чтобы стандарт не терял своей актуальности, на его основе должен развиваться некоторый практический базис – «скелет» современной среды компьютерного моделирования. Автором данной статьи был сформулирован план предлагаемого стандарта и разработан программный комплекс OpenMVLShell, который полностью согласуется с этим стандартом и реализует в себе на первой стадии ряд простых решений, которые обеспечивают поддержку всех этапов численного эксперимента.

В качестве языка моделирования был выбран язык Model Vision Language [8–10], который опирается на UML (стандарт в своей области) и развивает идею гибридного подхода к описанию моделей, который хорошо зарекомендовал себя на практике и был поддержан многими профессиональными пакетами компьютерного моделирования (Model Vision Studium, AnyLogic, Matlab, Dymola и др.).

Пакет компьютерного моделирования построен в соответствии с определяемой в стандарте структурой. Таким образом, поддерживается модульность оболочки и ее расширяемость. Каждый компонент отвечает за определенную фазу выполнения численного эксперимента и может быть независимо изменен при условии согласования его интерфейса с другими составляющими среды моделирования. Может быть реализовано несколько альтернативных модулей, выполняющих одни и те же функции различными способами. В таком случае можно проводить на практических тестах сравнение имеющихся программных решений по скорости построения и выполнения модели и по точности воспроизведения получаемого решения.

Для простоты модификации пакета Open MVL Shell было принято решение максимально упростить графический интерфейс и оставить в нем то необходимое, что достаточно для проведения всех этапов численного эксперимента:

1) редактор команд (загрузка модели в пакет, построение программной реализации и пр.);

2) испытательный стенд (временные и фазовые диаграммы).

Данное упрощение позволяет абстрагироваться от нюансов визуального отображения оболочки и сосредоточиться на решении фундаментальных проблем определенных в рамках стандарта. Несмотря на то, что такая минимизация сильно усложняет моделирование с точки зрения пользователя и делает Open MVL Shell непригодным для верстки сложных практических приложений, она делает данный программный комплекс простейшим подручным инструментом разработчика, которым можно воспользоваться для проверки каких-то решений и гипотез, сравнить их с существующими подходами.

Важно также, что проект Open MVL открыт к подключению для всех желающих [15]. Доступ к исходным кодам может получить любой программист, заинтересовавшийся разработкой и проблемами, имеющимися в моделировании. Таким образом, ставится цель в открытом формате накапливать исследовательский опыт посредством использования возможностей и ресурсов Интернета.

ПАКЕТ OPEN MVL SHELL В ОБУЧЕНИИ

На данной стадии развития проект поддерживается силами аспирантов и студентов кафедры РВКС Санкт-Петербургского Государственного Политехнического Университета (СПбГПУ). Аспиранты задействуют инструмент Open MVL Shell в своих диссертационных проектах. Студенты знакомятся с проектом Open MVL в рамках практических занятий по курсу «Среды визуального моделирования». Можно выделить два метода обучения практическому компьютерному моделированию:

1. «Пассивное» обучение (традиционное). Обучаемому студенту предлагается один (или несколько) пакетов компьютерного моделирования для воспроизведения на них некоторых модельных задач с заведомо известным решением. Подход к обучению относительно прост, и практический материал легко усваивается студентами. С другой стороны при этом может не достигаться глубокого понимания проблем объектно-ориентированного моделирования. Вдобавок к этому студент привыкает к конкретному графическому интерфейсу, опять же без понимания, какие проблемы ставятся перед пакетом, какие модели заведомо не могут быть решены, на каких моделях сложно ожидать «хорошего» решения и т. п.

2. «Активное» обучение. Из студентов собирается рабочий коллектив, которому предлагаются на выбор различные задачи, которые предстоит решить в рамках единого проекта – среды моделирования. Каждый студент знакомится с внутренним устройством пакета, с проблемами, которые автоматически решаются пакетом. Таким образом, каждый получает более глубокое понимание принципов объектно-ориентированного моделирования. Помимо этого, обучаемый студент имеет возможность реализовать свой творческий потенциал, взяв на себя решение задачи, не имеющей на текущий момент известного однозначно хорошего решения. Работа в коллективе над одним проектом дает опыт разработки в условиях, максимально приближенных к профессиональной разработке коммерческого приложения: задачи поставлены, сроки ограничены, требуется взаимодействовать с другими участниками группы, отлаживать реализованные решения, закрывая заявки в баг-трекинговой системе Redmine [16].

Именно активный метод обучения используется на базе проекта Open MVL. Данный подход, по мнению автора, наиболее эффективен с точки зрения обучения. Кроме того, не приходится получать учебные лицензии, которые требуются для использования почти всех профессиональных сред моделирования в рамках обучения.

На практических занятиях и семинарах в курсе «Среды визуального моделирования» рассказывается о потребности стандартизации в области моделирования [3–6], описывается предложенный прототип стандарта на разработку визуальных сред, обсуждается заявленная структура пакета, которая соответствует декларируемому стандарту, обозначаются проблемы, которые возникают при построении визуальных сред моделиро-

вания, и фиксируются направления исследования, которые можно провести на базе созданного инструмента.

Студенты начинают практическое обучение с ознакомления с документацией пакета, после чего даются задания построения различных моделей на языке объектно-ориентированного моделирования MVL. После нескольких занятий, когда основы объектно-ориентированного подхода усваиваются группой студентов на достаточном уровне, принимается решение, какие направления исследования задействовать в рамках текущего курса моделирования. Исследовательские задачи определяются с помощью системы управления проектами Redmine [16].

СТРУКТУРА OPEN MVL SHELL

В соответствии с декларируемой в рамках работы стандартом, пакет Open MVL Shell программно был разделен на следующий набор модулей:

– пользовательский интерфейс, работающий с пользователем в режиме командной строки (аналогично Open Modelica Shell [14]);

– командный компилятор, отрабатывающий команды пользователя, передающий в зависимости от команды управление другим подсистемам;

– компилятор модели на языке MVL;

– объектно-ориентированная база данных пакета;

– генератор программного кода построенной модели;

– подсистема, осуществляющая обработку математического описания, заданного пользователем, приведение ее к форме, используемой численными методами (форма Коши, x_d, x_a – дифференциальные и алгебраические составляющие в векторе состояния x, G_d, G_a – вещественные функции):

$$\begin{cases} x_d' = G_d(x_d, x_a, t) \\ 0 = G_a(x_d, x_a, t) \end{cases}, \quad x_d(0) = x_{d0}, \qquad (2)$$

$$x = \begin{bmatrix} x_d \\ x_a \end{bmatrix} \in \Re^n, \; t - \text{время};$$

– двигатель модельного поведения;

– численные методы;

– подсистема управления разделяемой памятью (*shared memory API*);

– подсистема рисования графиков – испытательный стенд.

Взаимодействие данных подсистем приведено на рис. 2. Двойной линией отмечены те составляющие, которые хранятся на жестком диске, сплошной – фиксированные компоненты пакета, пунктирной – динамически генерируемые в оперативной памяти составляющие проводимого численного эксперимента.

ТЕМЫ, ПРЕДЛАГАЕМЫЕ ДЛЯ ОБСУЖДЕНИЯ И ИССЛЕДОВАНИЯ

В проекте Open MVL предлагаются следующие темы для обсуждения:

1. Автоматическое построение модели

– новые алгоритмы поиска трансверсали, сравнение существующих алгоритмов;

– компонентное моделирование (важно для промышленных задач, когда становится необходимой группировка систем уравнений в блоки с «входами»/«выходами» или с «контактами»/«потоками» [7, 10]).

2. Преобразование и упрощение модели

– повышение производительности за счет определения структуры матрицы инцидентности (могут выявляться блоки, приводящие к возможности распараллеливания вычислений);

– символьное дифференцирование, которое позволяет понизить индекс системы, к которому чувствительны многие численные методы, предназначенные для решения алгебро-дифференциальных уравнений.

3. Численные методы

– символьное решение, которое может существенно понизить размерность системы;

– развитие итерационного подхода в численных методах на базе метода Крылова [1].

4. Автоматизация вычислительного эксперимента (создание системы тестирования и сравнения).

5. «Аккуратное» построение графиков (проблемы построения графиков обсуждались в статье [11]).

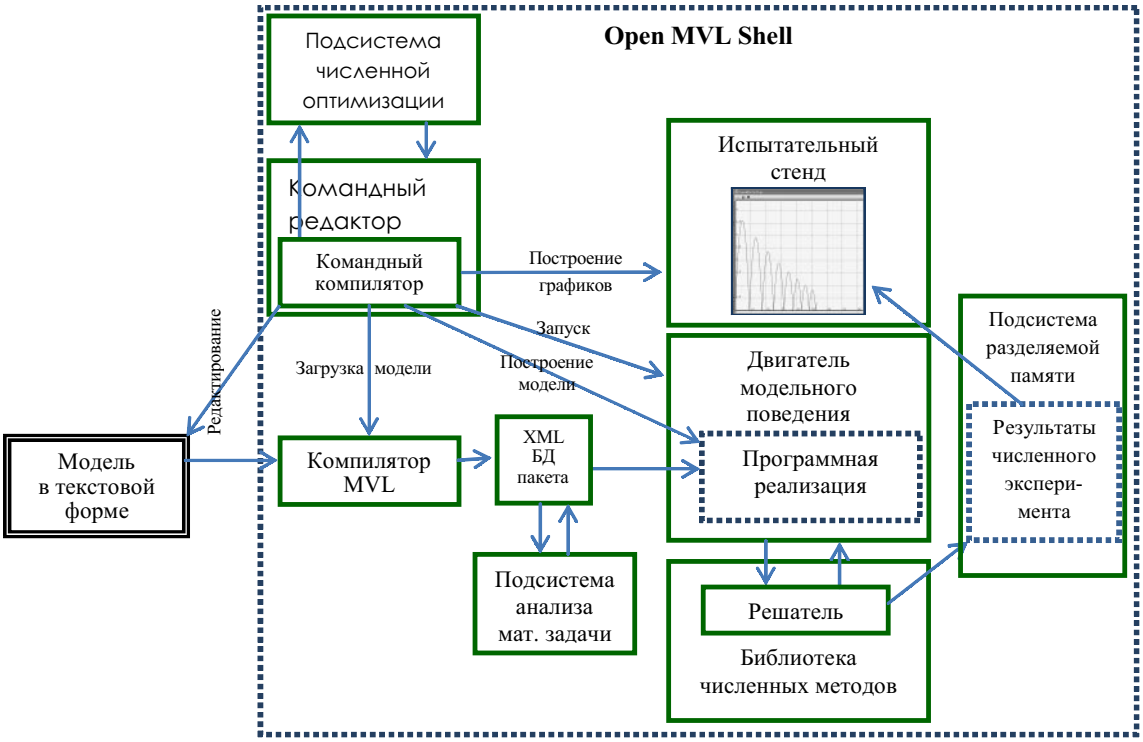

Рис. 2. Схема взаимодействия компонент в пакете Open MVL Shell

ПРИМЕРЫ РАСШИРЕНИЯ ВОЗМОЖНОСТЕЙ СРЕДЫ. СОЗДАНИЕ ИНСТРУМЕНТОВ МОДЕЛИРОВАНИЯ

Активные методы обучения могут быть реализованы в форме коллективной работы над проектом, «воспроизводящей» основные этапы научно-исследовательской работы: формулировка задачи, выбор методов решения, реализация, анализ и сравнение результатов. Новая научная монография или новый учебник являются прекрасным материалом для организации научно-исследовательской работы студентов:

– студентам (и преподавателям!) приходится самостоятельно изучать новый материал;

– известные или новые решения, предлагаемые автором книги, можно рассматривать как гипотезы, которые надо подтвердить или опровергнуть в процессе обучения;

– одним из способов подтверждения или опровержения гипотезы является проведение эксперимента (в нашем случае, вычислительного).

Одним из важнейших инструментов, необходимых для моделирования, является инструмент «Параметрическая оптимизация». Компонент, позволяющий найти оптимальные в заданном смысле параметры модели, существует в любой среде моделирования.

Перед студентами кафедры РВКС была поставлена задача разработать прототип компонента «Оптимизатор», используя методы, предлагаемые в новой монографии [12].

В результате в структуру оболочки Open MVL Shell был добавлен блок оптимизации, который позволяет при введении соответствующих команд в командной строке производить поиск минимума зааданного функционала (одномерная оптимизация, многомерная оптимизация, многокритериальная оптимизация).

ПРИМЕРЫ РАСШИРЕНИЯ ВОЗМОЖНОСТЕЙ СРЕДЫ. СОЗДАНИЕ КОНВЕРТОРА МОДЕЛЕЙ

Задача автоматического перевода текста описания моделей с одного языка модели-

рования на другой является чрезвычайно трудной задачей. Одним из способов решения этой задачи является двустадийный перевод текста. На первом этапе создается промежуточное описание модели, например на языке математики (восстанавливается математическая модель), а затем это описание переводится на требуемый язык моделирования.

Во многих средах построение математической модели происходит одновременно с ее преобразованием, связанным, прежде всего, с уменьшением числа решаемых уравнений или «улучшением» их численных свойств. Часто авторы среды даже не уведомляют пользователя о проведенных преобразованиях. Очевидно, что, если среда хранит в доступной форме информацию о решаемых уравнениях (в нашем случае речь идет об алгебро-дифференциальных уравнениях), конвертор может использовать эту информацию как исходную.

Построенные уравнения передаются Решателю. Качество и точность численного решения определяются многими факторами, в частности, алгоритмом продвижения модельного времени, алгоритмами поиска точки переключения (гибридные системы). Бессмысленно говорить о стандартизации процесса вычислений, так как конкретная среда строится, «затачивается» под решение определенного класса задач. Однако можно говорить о стандартных для всех сред процедурах, нужных только для проверки правильности конвертирования. Это может быть стандартный модуль, с унифицированным интерфейсом, подключаемый к любому пакету. В простейшем случае – это унифицированная численная библиотека и алгоритм продвижения модельного времени (календарь событий).

В нашем конкретном случае, исходным пакетом был выбран пакет Simulink. Файл, содержащий описание модели, подается на вход Конвертора. Стандартная библиотека Конвертора содержит большинство блоков пакета Simulink, реализованных на языке MVL. Эти модели были разработаны вручную и представляют собой открытые гибридные автоматы, соответствующие блокам Simulink. Конвертор распознает блоки на исходном языке, «настраивает» их и соединяет в структурную схему, соответствующую структурной схеме исходной модели.

Первые эксперименты показали, что для простейших (структурно простых) моделей можно добиться совпадения численных результатов на отдельных траекториях (гибридные системы). Как сравнивать и гарантировать совпадение на всем допустимом множестве траекторий для конкретных моделей, остается открытым. Однако и этот результат является важным на начальном этапе разработки Конверторов.

ЗАКЛЮЧЕНИЕ

Ключевым для данного проекта является требование открытости на уровне исходных кодов. Открытость кодов позволяет подключать к совместной работе неограниченное число разработчиков по всему миру, воспроизводить предлагаемые решения, сравнивать результаты.

В перспективе ставится цель создать стандарт на разработку среды визуального моделирования.

Таким образом, используя возможности Интернета, к исследованиям в проекте Open MVL можно подключать все большее число разработчиков, каждый из которых может модифицировать только один или несколько блоков, сконцентрировав усилия над фундаментальными математическими проблемами разработки пакетов компьютерного моделирования.

Проект Open MVL аналогичен операционной системе Linux: открытые исходные коды позволяют привлечь максимальное число людей к исследовательской работе над актуальными проблемами в изучаемой предметной области – моделировании.

Литература

1. *Peter N.B., Hindmarsh A.C., Petzold L.R.* Using Krylov Methods in the Solution of Large-Scale Differential-Algebraic Systems, 1993.

2. *Popper N., Breitenecker F.* Extended and Structural Features of Simulators. Simulation News Europe, 2008. Dec. P. 27–38.

3. *Исаков А.А., Сениченков Ю.Б.* Среда визуального моделирования Open MVL / XXXIX Неделя науки СПбГПУ. Материалы международной научно технической конференции. СПб.: Изд-во Политехнического Университета, 2010. С. 151–153.

4. *Исаков А.А., Сениченков Ю.Б.* Среда визуального моделирования Open MVL / Вычислительные, измерительные и управляющие системы. СПб.: Изд-во Политехнического Университета, 2010. С. 84–89.

5. *Исаков А.А., Сениченков Ю.Б.* Program Complex Open MVL for Modeling Complex Dynamic Systems // Электронный журнал «Дифференциальные Уравнения и Процессы Управления», Математико-механический факультет Санкт-Петербургского Государственного университета, 2011.

6. *Исаков А.А.* Open MVL Shell – исследовательская среда визуального моделирования // Всероссийский конкурс научно-исследовательских работ студентов и аспирантов «Инновационные технологии в образовательном процессе», Белгородский государственный национальный исследовательский университет, 2011. С. 11–15.

7. *Колесов Ю.Б., Сениченков Ю.Б., Беньковић Е.С.* Практическое моделирование динамических систем. СПб.: БХВ-Петербург, 2002.

8. *Колесов Ю.Б., Сениченков Ю.Б.* Моделирование систем. Объектно-ориентированный подход. СПб.: БХВ-Петербург, 2006.

9. *Колесов Ю.Б., Сениченков Ю.Б.* Моделирование систем. Динамические и гибридные системы. СПб.: БХВ-Петербург, 2006.

10. *Колесов Ю.Б., Сениченков Ю.Б.* Моделирование систем. Практикум по компьютерному моделированию. СПб.: БХВ-Петербург, 2007.

11. *Сениченков Ю.Б.* Численное моделирование гибридных систем. СПб.: Изд-во Политехнического университета, 2004.

12. *Черноруцкий И.Г.* Методы оптимизации. Компьютерные технологии. СПб.: БХВ-Петербург, 2011.

13. Официальный сайт проекта Open Modelica / http://www.openmodelica.org (дата обращения 29.10.2012).

14. Проект Open MVL в Интернете / http://dcn.ftk.spbstu.ru/index.php?id=276; https://dcn.ftk.spbstu.ru/redmine/projects/openmvl (дата обращения 29.10.2012).

15. Официальный сайт Redmine / http://www.redmine.org (дата обращения 29.10.2012).

Abstract

*OpenMVL*projectis consideredas the starting point for discussion and the first realized step in the direction of building new standard for modeling and simulation of complex dynamical systems tools. OpenMVLShell is open source software demanding for comparison different approaches to designing modern environments for modeling and simulation first of all. However it may be used in Education too as its analog OpenModelica. OpenMVLShell is based on Object-Oriented Modeling (OOM) approach and uses Model Vision Language (MVL) suggested by MvStudiumGroup.

Keywords: simulators, standards, package design, simulation languages, object modeling techniques, problem solvers.

Исаков Андрей Алексеевич,
магистр кафедры РВКС СПбГПУ
(аспирант второго года обучения),
isak239@mail.ru.