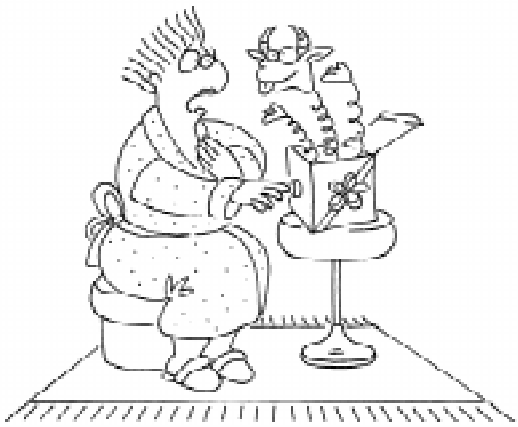
Инихов Дмитрий Борисович,
Колесов Юрий Борисович,
Сениченков Юрий Борисович

«ФИЗИЧЕСКОЕ» МОДЕЛИРОВАНИЕ В MVSTUDIUM

Кто из нас в детстве не играл в «кубики»? Все играли. Строили домики, крепости, складывали из фрагментов картинки. Кто остался верен этой игре на всю жизнь и в зрелом возрасте складывает из «кубиков» поразительные по сложности конструкции? Например, теоретики и практики технологического подхода к производству товаров массового потребления. Строители, производители стиральных машин и автомашин, и любой другой сложной бытовой техники.

Однако этот подход успешно работает и в современных наукоемких областях. Вычислительные машины и современные программы тоже собирают из кубиков.

Программирование в эпоху рождения вычислительных машин справедливо относилось к искусствам – первые программисты из нулей и единичек, объединенных в команды, создавали достаточно сложные программы, которые успешно работали! Ну чем не вышивание бисером! Массовый спрос на программное обеспечение разделил программистов на «художников» и «ремесленников». «Художники» создают уникальные кубики, а ремесленники из готовых «кубиков» создают архисложные программы, среди которых встречаются и произведения искусства.



Если представить объект в виде «черного» ящика...

Компьютерное моделирование сложных динамических систем сегодня стало основой проектирования и, конечно, тоже использует блочный подход при разработке моделей. Различают два типа блоков – «ориентированные» блоки, или блоки с входами и выходами и «неориентированные» блоки, или блоки с контактами. Напомним, что блоки с входами и выходами пришли к нам из теории управления. Запишем закон поведения объекта управления в виде

$$\frac{ds}{dt} = f(s(t), x(t)),$$

$$y(t) = g(s(t), x(t)),$$

где $s(t)$ – состояние объекта, $x(t)$ – заданная функция управления, $y(t)$ – наблюдаемая величина. Если представить объект в виде «черного» ящика, то на его вход поступает сигнал $x(t)$, а на его выходе мы видим функцию $y(t)$. Естественно функцию $x(t)$ назвать «входом», а функцию $y(t)$ – «выходом». Отсюда и еще одно часто употребляемое название – ориентированные блоки.

Блоки с контактами наиболее часто применяются при моделировании электрических, механических систем и вообще любых других систем, для которых применимы аналоги законов Кирхгофа.

Рассмотрим конкретный пример (своеобразная «визитная карточка» авторов программного продукта Modelica, использовавших его для иллюстрации преимуществ своего продукта).

Пусть мы имеем два готовых устройства – электродвигатель постоянного тока M и пропеллер P – и хотим построить систему «вентилятор», соединив их жестким валом (рис. 1).

В данном примере мы имеем два очевидных блока — двигатель M и пропеллер P, поведение которых упрощенно задается следующими уравнениями:

$$\text{(M)}\ \begin{cases} \dfrac{d\omega}{dt} = \dfrac{k_1 I + T}{J_a}, \\ \dfrac{dI}{dt} = \dfrac{V - k_2\omega - R_a I}{L_a}, \end{cases}$$

$$\text{(P)}\ \begin{cases} \dfrac{d\omega}{dt} = \dfrac{T - k_p\omega}{J_p}, \end{cases}$$

где ω, T – угловая скорость и момент на валу двигателя и пропеллера, V – напряжение на двигателе, I – ток якоря, J_a, R_a, L_a – момент инерции, омическое и индуктивное сопротивление якоря, соответственно, J_p – момент инерции пропеллера, k_1, k_2, k_p – специальные коэффициенты. Очевидно, что переменные $M.V$, $M.\omega$, $M.T$ являются внешними для блока M, а переменные $P.\omega$, $P.T$ – для блока P в том смысле, что мы имеем модели «мотора, вращающего нагрузку», и «нагрузки, вращаемой мотором». И если задать коэффициенты дифференциальных уравнений мотора и нагрузки, то мы получим классические динамические системы независимых физических объектов.

Теперь попробуем соединить эти блоки в единое устройство, имитируя процесс создания вентилятора из двух готовых типовых элементов. При соединении двигателя и пропеллера абсолютно жестким валом возникают еще два уравнения:

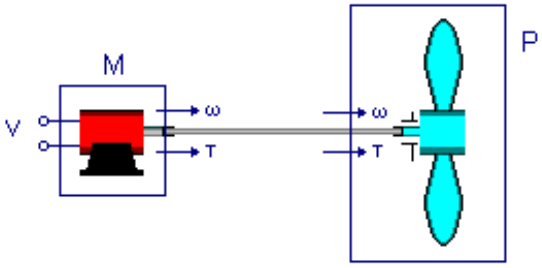


Рис. 1. Соединение мотора и пропеллера в единую систему «вентилятор»

$$\begin{cases} M.\omega = P.\omega, \\ M.T + P.T = 0. \end{cases}$$

Посмотрим на эти пять уравнений внимательно. Величины $M.\omega$, $P.\omega$ одновременно удовлетворяют двум дифференциальным уравнениям, так как они являются переменными состояния для каждого из блоков и удовлетворяют каждая своему дифференциальному уравнению, в которые также входят величины $M.T$ и $P.T$, связанные соотношением $M.T = -P.T$. Они также должны быть равны друг другу, так как соединены дополнительной связью. Следовательно, физическая связь между двигателем и пропеллером не может быть описана с помощью ориентированной связи, принятой в традиционном блочном моделировании, а сами эти физические блоки не являются ориентированными блоками. В рамках подхода, использующего блоки с входами и выходами, эта система может быть описана только в виде *одного* ориентированного блока, то есть вентилятор целиком, где управляющее напряжение – «вход», а угловая скорость вращения – «выход»:

$$\begin{cases} \dfrac{d\omega}{dt} = \dfrac{k_1 I - k_p\omega}{J_a + J_p}, \\ \dfrac{dI}{dt} = \dfrac{V - k_2\omega - R_a I}{L_a}. \end{cases}$$

Для получения приведенного описания в виде системы уравнений первого порядка необходимо рассмотреть два «одинаковых» уравнения,

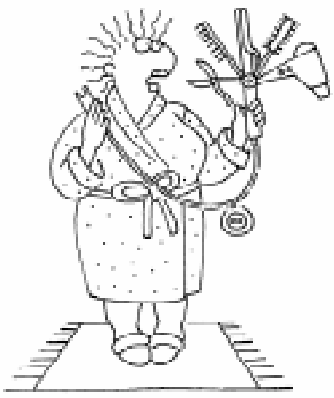
$$\begin{cases} \dfrac{d\omega}{dt} = \dfrac{k_1 I + T}{J_a}, \\ \dfrac{d\omega}{dt} = \dfrac{-T - k_p \omega}{J_p}, \end{cases}$$

исключить из них T (знак T во втором уравнении изменен с учетом уравнения $M.T + P.T = 0$) и получить искомую систему.

Ориентированными, как правило, можно считать лишь искусственные технические блоки, в которых инженеры приложили усилия для исключения обратного влияния (например, поставили соответствующие развязки). В физических системах такие развязки могут и отсутствовать. Авторы Modelica справедливо отмечают, что использование только однонаправленных связей и ориентированных блоков (причинно-следственный подход) практически закрывает возможность создания библиотек типовых блоков для различных прикладных областей и, следовательно, препятствует широкому использованию математического моделирования в современном инженерном деле.

Было время, когда эти подходы противопоставляли, но, как всегда бывает, в итоге осознали, что они естественным образом дополняют друг друга. В современных программных продуктах оба подхода прекрасно уживаются, давая возможность наилучшим образом использовать специфику задачи.

Семейство программных продуктов MvStudium (www.mvstudium.com) – не исключение. Последняя версия продукта MvStudium 5.0 тоже позволяет использовать блоки с контактами. В то же время в MvStudium есть еще одна важная особенность использования как блоков с контактами, так и с блоков с входами–выходами. В MvStudium и те, и другие могут менять свое поведение, в зависимости от происходящих событий. Пользуясь простейшими аналогиями, можно пояснить это так. Предположим, что вам нужен набор инструментов для различного вида работ, например отверток. Раньше вы покупали набор из десяти различных отверток, а теперь – одну отвертку со сменными насадками. В этом и состоит отличие MvStudium от других систем моделирования. Блоки по ходу дела можно перенастраивать, меняя системы уравнений, определяющих их поведение. Число различных систем уравнений (поведений) в одном блоке, конечно, заранее предопределено и на практике не очень велико. Но даже при небольшом числе вариантов поведения в одном блоке при соединении таких блоков в систему число вариантов итоговой, совокупной системы может стать очень большим, и любой из этих вариантов, вообще говоря, может когда-нибудь реализоваться. Не создавать же их заранее, и не хранить же их все в памяти машины в виде готовых к исполнению программ! Вариантов слишком много, а в конкретной ситуации реализуется только часть возможных комбинаций. К счастью, для блоков с входами и выходами достаточно хранить только отдельные поведения (фрагменты кода, их реализующие) – совокупное поведение получается механическим объединением заранее подготовленных фрагментов. Рассмотренный выше пример вентилятора показывает, что при использовании блоков с контактами необходимо преобразовывать уравнения отдельных блоков, чтобы получить совокупную систему. Эти преобразования нельзя выполнить заранее! Сейчас в MvStudium они выполняются по ходу решения, когда возникает необходимость. Это замедляет процесс моделирования, но будем надеяться, что не очень сильно.



Вариантов слишком много, а в конкретной ситуации реализуется только часть возможных комбинаций.

Сейчас мы не будем обсуждать событийно-управляемые системы и предположим, что речь идет только о блоках с неизменяемым поведением.

Для демонстрации новых возможностей пакета выберем наиболее выигрышный пример – электрические схемы. В чем его преимущество перед другими? Главное преимущество электрических схем в данном случае в том, что существует наглядный язык представления схем, понятный практически всем. В других областях, например в механике, хотелось бы, чтобы будущая система собиралась из компонентов аналогичным образом. Пусть мы строим математическую модель маятника, нарисованного в виде стержня, один конец которого закреплен, а на другом находится сосредоточенная масса (графический язык пакета). Не всякий современный пакет моделирования может предоставить вам такой графический редактор, с помощью которого вы сможете соединять предопределенные примитивы в единое целое, чтобы потом автоматически построить нужные уравнения. Еще относительно недавно в качестве входных графических языков для механических, гидравлических и других аналогичных систем предлагались весьма сложные графические конструкции, распознать в которых исходный моделируемый объект было достаточно трудно [2]. Один из возможных путей – взять готовый промышленный редактор, умеющий только рисовать, научиться анализировать его текстовый язык, чтобы извлекать нужные исходные данные и передавать их в блок пакета, умеющего строить уравнения.

В электрических схемах вполне допустимо использовать слегка модифицированные функциональные схемы (рис. 2), которые легко воспринимаются как электрические схемы.

Графический язык пакета позволяет внутри прямоугольников размещать иконки, и, если захотеть, схема будет очень похожа на схемы из учебников.

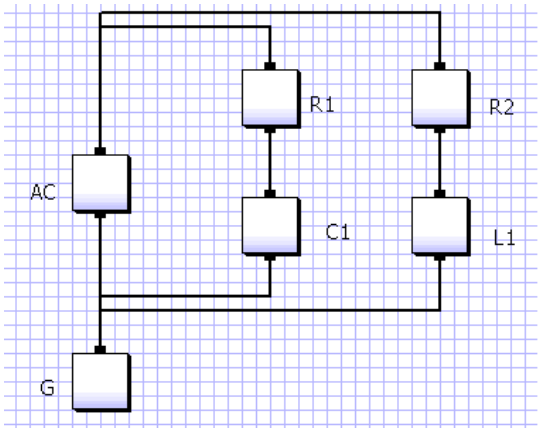


Рис. 2. Достаточно легко читаемая электрическая схема. Пример окна нового редактора функциональных схем MvStudium 5

Знакомые с работой предыдущих версий MvStudium обратят внимание на то, что на функциональных схемах появился новый элемент – контакт. С точки зрения электрических схем контакты обеспечивают со-

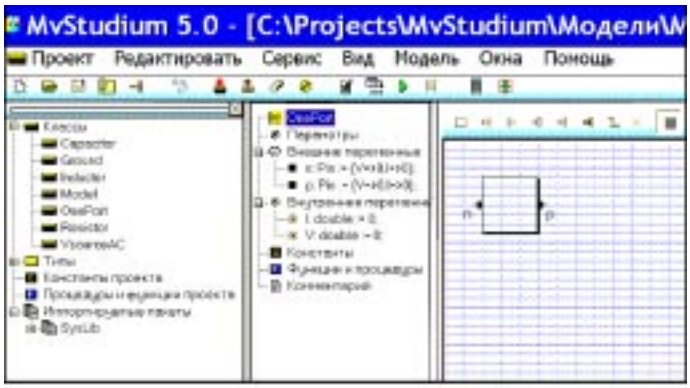


Рис. 3. Двухполюсник. Интерфейс

Рис. 4. Двухполюсник. Уравнения

единение различных компонентов: источников питания, сопротивлений, индуктивностей, емкостей. В свою очередь, эти компоненты могут быть представлены как модификации (наследники) абстрактного двухполюсника (рис. 3–4).

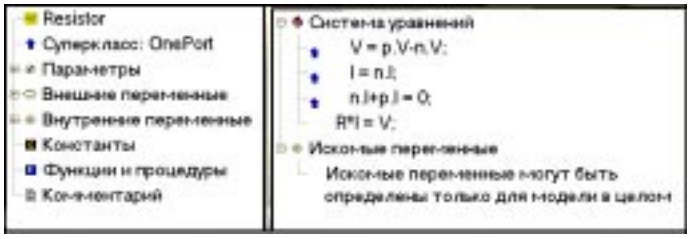


Рис. 5. Класс «Резистор»

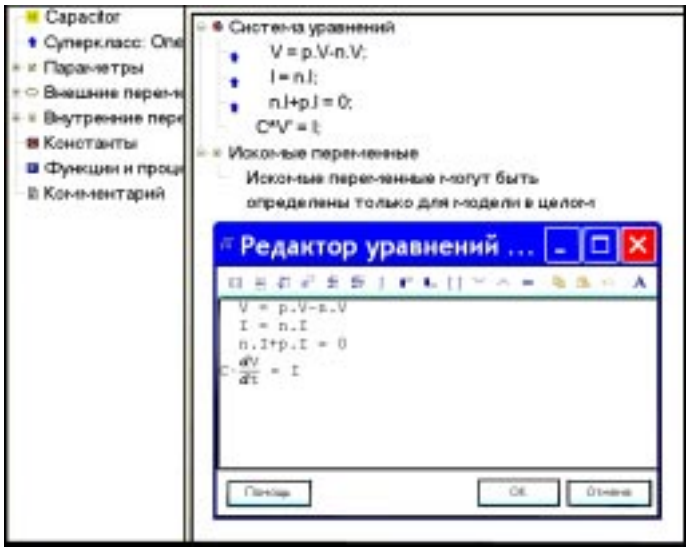


Рис. 6. Класс «Емкость»

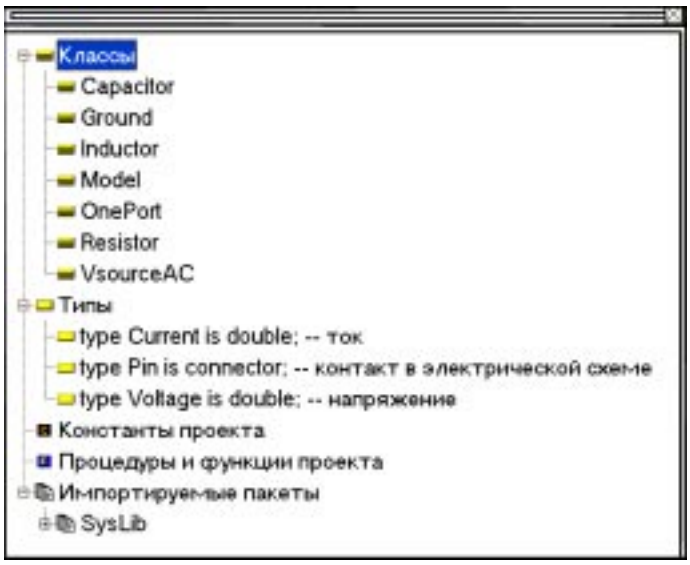


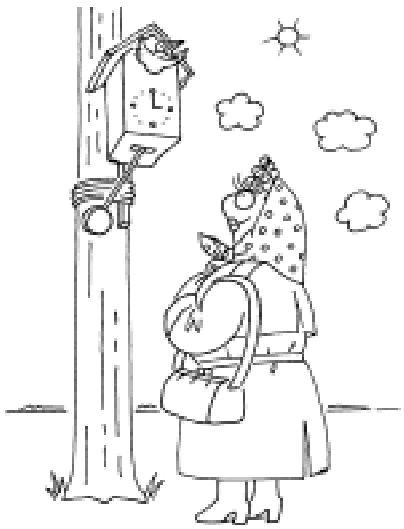
Рис. 7. Пример проекта
с пользовательскими типами данных

Наделяя абстрактный двухполюсник конкретными свойствами (компонентными законами или заданным поведением), можно построить модели сопротивлений (рис. 4), емкостей (рис. 5) или других нужных деталей электрических схем.

Введение блоков с контактами потребовало новых типов данных (рис. 7). Это так называемые пользовательские типы данных, в основе которых лежат структуры.

Как нельзя кстати оказался и механизм наследования. Стрелки на рис. 5–6, расположенные рядом с переменными, говорят о том, что переменные унаследованы от родителя блоков – двухполюсника. Собственное поведение определяется компонентными законами – для класса «Резистор» это закон Ома.

Класс «Модель», имеет компонентную структуру, и в нем размещается собственно электрическая схема (рис. 2). Электрическая схема задает так называе-



...строим математическую модель маятника...

мые топологические уравнения, которые вместе с компонентными уравнениями используются для построения совокупной системы. Решение этой системы мы видим на рис. 8.

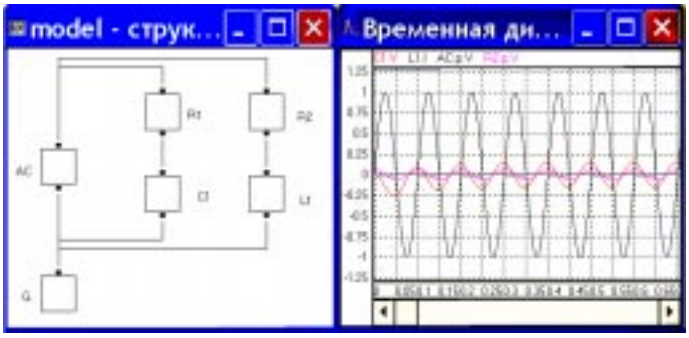


Рис. 8. Испытательный стенд с функциональной схемой и временной диаграммой

При использовании блоков с контактами пользователь может не задумываться ни о компонентных уравнениях (их заранее написали авторы библиотеки компонентов), ни о топологических – они получаются автоматически, после того как схема нарисована. Моделирование становится похожим на сборку устройства из готовых «физических» блоков: выбрал компоненты, соединил их, включил – работает. Отсюда и название – «физическое» моделирование. Этот подход очень часто хвалят те, кто справедливо отмечает, что представление модели в виде абстрактной системы уравнений сложно для восприятия, точно так же, как сложно воспринимать модель, записанную на языке программирования. В основе инженерных языков и методов лежат чаще всего интуитивные модели, и языки блочного моделирования возвращают нас к старым привычным схемам.

До выхода в свет пользовательской версии осталось недолго – авторы надеются что уже этой весной ее можно будет найти на нашем сайте.

Литература

1. *Арайс Е.А., Дмитриев В.М.* Автоматизация моделирования многосвязных механических систем. «Машиностроение», 1987. 240 с.

2. *Корячко В.П., Курейчик В.М., Норенков И.П.* Теоретические основы САПР. «Энергоатоиздат», 1987. 399 с.

Инихов Дмитрий Борисович,
зам. директора фирмы «MvSoft»,

Колесов Юрий Борисович,
доктор технических наук,
профессор кафедры РВКС
факультета технической
кибернетики СПбГПУ,

Сениченков Юрий Борисович,
доктор технических наук,
профессор кафедры РВКС
факультета технической
кибернетики СПбГПУ.