ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД К РАЗРАБОТКЕ ТРЕНАЖЕРА ГРУЗОВЫХ ОПЕРАЦИЙ НА МОРЕ

Д. В. Киптилый, Ю. Б. Колесов, Д. В. Лебедев, Ю. Б. Сениченков, С. В. Тарасов
(Санкт-Петербург)

1. Введение

Построение тренажёра современного грузового судна [1] подразумевает создание и объединение математических моделей судовых технологических систем. В их число входят такие системы как балластная, грузовая, система инертного газа, пожаротушения, очистки грузовых танков и т. п. Эти системы состоят из большого количества элементов. Например, грузовые системы танкеров-перевозчиков химикатов могут состоять из нескольких десятков танков, сотен труб и управляющих клапанов. Математическое моделирование подобных систем в реальном времени является достаточно сложной задачей и требует особого подхода. Сложность математического описания динамики поведения рассматриваемых систем заключается в том, что эта динамика определяется несколькими тысячами нелинейных дифференциально-алгебраических уравнений переменной структуры. Логика изменения системы уравнений определяется тысячами условий. Построение такой модели вручную – задача практически невыполнимая. Единственным выходом является использование компонентного моделирования [2]. Современное компонентное моделирование является объектно-ориентированным [3]. При использовании этого подхода проводится объектно-ориентированный анализ прикладной области, в результате которого выделяются типовые классы и отношения наследования между ними. Определения этих классов на математическом уровне составляют библиотеки типовых компонентов. Описание структуры модели и типовых компонентов приводится в терминах языка UML среды разработки математической модели. Конкретная модель собирается из этих типовых компонентов, соединенных связями. Совокупное поведение всей модели в целом воссоздается исполняющей системой, входящей в состав инструмента моделирования. Для использования модели в тренажёре необходимо создать выполняемый программный код. Это код генерируется инструментом моделирования по математическому описанию системы и включается в состав программного обеспечения тренажёра. Описанный подход соответствует технологии «проектирование на базе моделирования», декларированной корпорацией MathWorks [4]. Именно такая технология была внедрена в разработку морских технологических тренажёров компанией «Транзас». В качестве средства разработки многокомпонентных моделей была выбрана среда моделирования «Model Vision Studium» [5].

В настоящей статье детально описывается использованный подход к построению математических моделей многокомпонентных систем. В разделе 2 мы опишем архитектуру и продемонстрируем математическое содержание типичной модели многокомпонентной системы. В разделе 3 будет дано описание среды разработки.

2. Архитектура модели

2.1. Описание математической модели

Математическими моделями тренажеров в нашем случае являются сложные динамические (гибридные) системы, поведение которых зависит от событий (в каждый момент времени решается одна из множества систем алгебро-дифференциальных уравнений большой размерности, порождаемого гибридным автоматом). Так как модель строится из большого числа типовых компонентов, ис-

пользуется объектно-ориентированный подход [2]. Каждый компонент тренажера обычно содержит свой собственный гибридный автомат (карту поведения [2]). Таким образом, математические модели тренажеров представляют собой параллельную композицию большого числа гибридных автоматов. Согласование работы гибридных автоматов компонентов осуществляется с помощью *принципа синхронной композиции* [2]. В моделях используется *явная синхронизация* гибридных автоматов, осуществляемая с помощью *механизма сигналов* - сигналы посылаются всем гибридным автоматам компонентов при наступлении дискретного события в одном из них [3]. Прикладная область (гидродинамика, аэродинамика, управление) предполагает использование компонентов с внешними переменными типа «контакт-поток» («физическое» моделирование) и переменными типа «вход-выход». Использование технологии «физического» моделирования при построении тренажеров приводит к необходимости строить текущую систему алгебро-дифференциальных уравнений во время исполнения модели, что обычно трудно совместимо с требованием моделировать в реальном времени. Указанное противоречие удалось преодолеть, так что создаваемые тренажеры, не смотря на большое число уравнений, являются моделями, работающими в реальном времени.

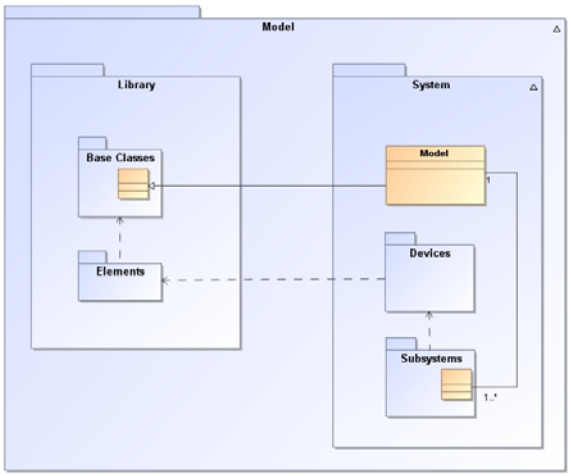


Рис. 1 Общая структура модели Model

2.2. Описание проекта модели

Проект Model математической модели сложной динамической системы состоит из: 1) внешнего пакета элементарных компонентов Math_Hydraulics, 2) пакета устройств системы Devices, 3) пакета подсистем Subsystems, реализующих полный функционал системы и 4) объединяющего класса Model. Каждый пакет состоит из набора классов. Общая структура модели показана на Рис. 1. Рассмотрим по-отдельности составные части модели.

Пакет элементарных компонентов Math_Hydraulics состоит из: a) пакета абстрактных базовых классов и b) пакета классов, моделирующих элементарные функциональные составляющие системы (такие как: трубы, клапаны, ёмкости, насосы и т. п.).

Пакет устройств Devices системы представляет собой набор классов, моделирующих отдельные технологические узлы системы (например, танк с насосом, нагревателем, прилегающими трубами и управляющими клапанами).

Пакет подсистем Subsystems является набором классов, объединяющих модели различных устройств в подсистемы моделируемой системы (например, группа грузо-

вых танков, входящая в состав системы Cargo или подсистема генерации инертного газа, являющаяся составной частью системы InertGas).

Наконец, *класс* Model a) объединяет все подсистемы в единое целое, и b) управляет поведение устройств с помощью сигналов. Этот класс имеет свою карту поведения.

Эти составляющие модели будут подробно рассмотрены ниже.

2.3. Пакет элементарных компонентов Math_Hydraulics

Классы, входящие в этот пакет можно разбить на две группы: абстрактные базовые классы и классы функциональных элементов. Пакет содержит также общие константы, типы данных, специфические для решаемых задач, и общие для всех классов процедуры и функции.

2.3.1. Константы и типы данных

В библиотечном пакете определяются пользовательские типы данных, удобные для решаемых задач гидро- и аэродинамики. Например, для создания математической модели трубопровода используются экземпляры класса Pipe. Этот класс имеет две группы внешних переменных, описывающих свойства жидкости на обоих концах моделируемого потока вещества в трубе, такие как: давление, расход, температуру, вектор концентраций и другие. Их удобно объединить в единый тип данных **liquid**:

```
type liquid is
  connector
    contact  P:    double; -- pressure [bar]
    flow     Q:    double; -- volume flow [m3/s]
    contact  T:    double; -- temperature [K]
    contact  conc: vector; -- concentrations [%]
    …              -- other functional and auxiliary variables
end connector;
```

И тогда, например, в классе PipeBase можно использовать только два внешних коннектора inlet и outlet: connector inlet: liquid;connector outlet: liquid; описывающих характеристики вещества на обоих концах трубы.

Другим примером пользовательского типа данных может быть набор параметров data_pipe, характеризующих свойства трубы (высоты концов трубы, диаметр, длина, коэффициент трения):

```
type data_pipe is
  record
    height_in:  double; -- inlet height [m]
    height_out: double; -- outlet height [m]
    diam:       double; -- pipe diameter [m]
    len:        double; -- pipe length [m]
    dzeta:      double; -- coefficient of local hydrodynamic loss
end record;
```

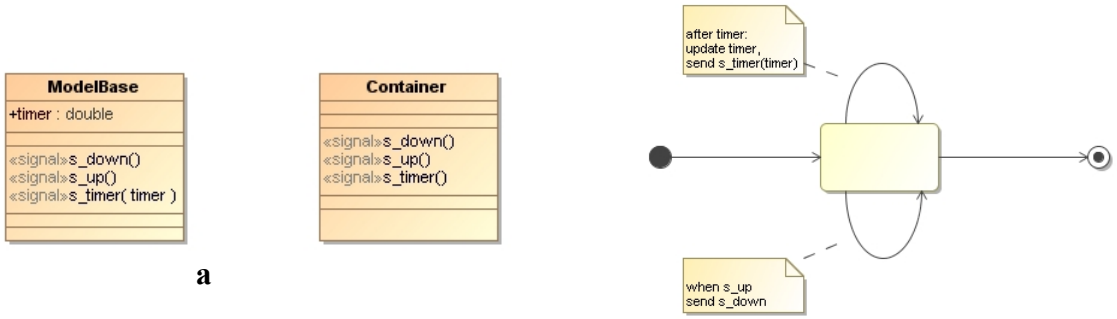
Класс PipeBase содержит группу параметров Data_Pipe: parameter Data_Pipe: data_pipe;

2.3.2 Базовые классы

В число базовых классов входят такие классы как ModelBase, Container, PipeBase, ValveBase, TankBase и т. д. Эти классы содержат базовые константы, параметры, переменные (внешние и внутренние) и методы (процедуры и функции), которые в дальнейшем используются наследниками.

Каждый класс библиотеки (за исключением управляющих классов ModelBase и Controller) является наследником класса Container, в котором реализован механизм передачи сигналов. Как уже говорилось, этот механизм необходим для явной синхронизации всех гибридных автоматов модели.

В моделях реализованы два типа сигналов: сигналы состояния (s_up и s_down) и сигнал времени (s_timer). Сигналы состояния передаются для синхронизации состояний всех элементов модели, сигналы времени для синхронизации дискретных действий (например, обновления переменных индикаторов). Механизм управления сигналами реализован в классе ModelBase (Рис. 2.а). Упрощённая карта поведения класса ModelBase схематично изображена на Рис. 2.б (класс Container не имеет своей карты поведения).



а

б

Рис. 2 а) Классы ModelBase и Container и

б) Карта поведения класса ModelBase

2.2.3 Классы функциональных элементов

Каждый класс функционального элемента (за исключением управляющего класса Controller) является наследником одного из базовых классов. Например, класс Pipe, модель трубы, является наследником класса PipeBase: Pipe → PipeBase; класс Valve (труба с управляющим клапаном) наследует класс ValveBase: Valve → ValveBase.

Каждый класс содержит иерархическую *карту поведения*. Каждое состояние такой карты является либо простым, либо иерархической картой поведения. Простым состояниям приписывается *непрерывное поведение,* описываемое системой уравнений, остальным – гибридное. Непрерывные и гибридные поведения реализуются в форме *локальных классов.*

Для иллюстрации вопросов, обсуждаемых в этом разделе, мы будем использовать библиотечный класс Pipe, моделирующий течение жидкости или газа по прямолинейной трубе круглого сечения.

Гибридный класс OpenedMap используется для описания течения жидкости по трубе. Его карта поведения изображена на Рис. 3 а.

Карта поведения OpenedMap содержит одно вспомогательное состояние fix_transfer и несколько функциональных. Для удобства моделирования мы разделили процесс течения жидкости на пять состояний-режимов: no_flux (поток отсутствует),

lam_pos (прямое ламинарное течение), turb_pos (прямое турбулентное течение), lam_neg (обратное ламинарное течение), turb_neg (обратное турбулентное течение).

Поведение системы в каждом из этих состояний описывается своим локальным непрерывным классом.

В классе Pipe в ламинарном режиме течения по горизонтальной трубе (состояние lam_pos Рис. 3 а), в простейшем случае, решается следующая система уравнений для давлений и потоков:

$$\begin{aligned} &inlet.P - outlet.P = k_{lam}\, inlet.Q, \\ &inlet.Q + outlet.Q = 0 \end{aligned} \qquad (1)$$

Здесь k_{lam} - известный коэффициент, характеризующий трубу и ламинарный режим течения. Для более полного описания течения, система дополняется уравнениями, описывающими состав и термодинамические свойства вещества.

Заметим, что для решения приведённой системы уравнений на неё должны быть наложены два дополнительных условия. Эти условия задаются другими элементами модели (внешними по отношению к элементу Pipe). Например, мы можем задать манометрические давления на разных концах трубы:

$$\begin{aligned} &inlet.P = P_0 > 0, \\ &outlet.P = 0 \end{aligned}$$

Приведённая система составляет тело локального непрерывного класса. Известные величины указываются при помощи оператора **known**:

```
inlet.Q + outlet.Q = 0;
inlet.P - outlet.P = k_lam * inlet.Q;
known k_lam;
```

В непрерывном классе также можно определить переменные и методы для формирования системы уравнений.

Заметим здесь, что соединение нескольких труб приведёт к совместному решению нескольких систем уравнений, дополненных уравнениями связи между трубами. Приве дём для наглядности полную систему уравнений для модели ламинарного течения жидкости в двух последовательно соединённых горизонтальных трубах (Pipe1 и Pipe2).

Pipe 1:

Pipe1.*inlet.P* – **Pipe1.***outlet.P* = $k_{lam}^{(1)}$ **Pipe1**.*inlet.Q*,

Pipe1.*inlet.Q* + **Pipe1.***outlet.Q* = 0;

Pipe 2:

Pipe2.*inlet.P* – **Pipe2.***outlet.P* = $k_{lam}^{(2)}$ **Pipe2.***inlet.Q*,

Pipe2.*inlet.Q* + **Pipe2.***outlet.Q* = 0;

Связи:

Pipe1.*outlet.P* = **Pipe2.***inlet.P*,

Pipe1.*outlet.Q* + **Pipe2.***inlet.Q* = 0;

Дополнительные условия:

Pipe1.*inlet.P* = P_0 > 0,

Pipe2.*outlet.P* = 0.

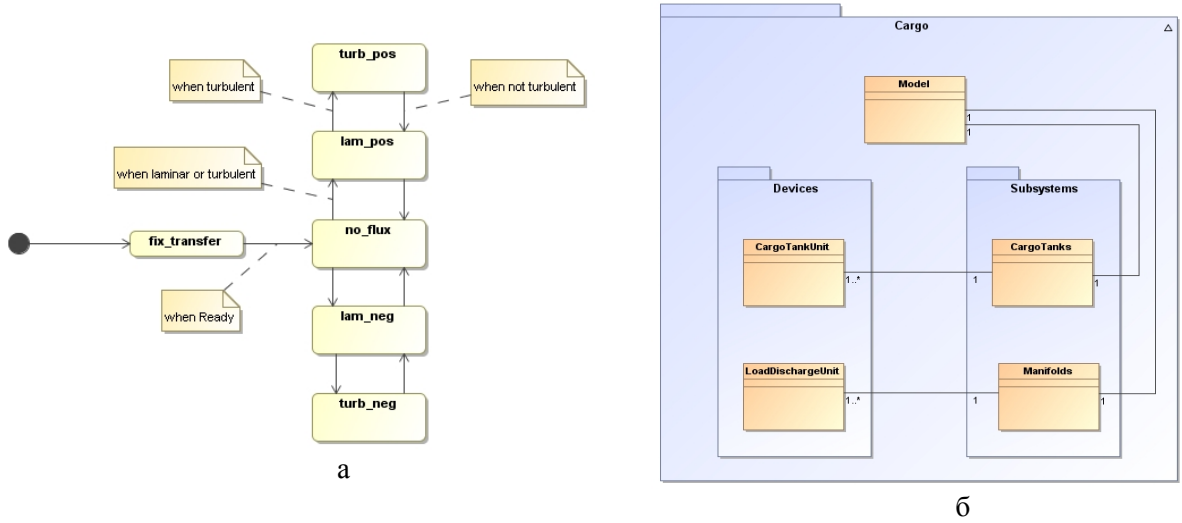
Дополнительные условия являются результатом решения уравнений во внешних (по отношению к Pipe1 и Pipe2) классах и уравнений связей между ними. Для простоты мы опускаем детальное изложение этого вопроса.

Рассмотренный здесь принцип построения математической модели трубы с жидкостью является общим для всех классов.

2.4 Пакет устройств и подсистем System

В пакет System импортируется пакет библиотечных классов Math_Hydraulics. Пакет System содержит явным образом пакет устройств Devices и подсистем Subsystems (Рис. 1). Все классы этих пакетов являются наследниками класса Math_Hydraulics.Container, отвечающего за механизм передачи сигналов.

Опишем структуру пакета на примере математической модели грузовой системы Cargo нефтеналивного танкера. Эта система (Рис. 3 б) состоит из двух подсистем (подсистемы танков CargoTanks, и подсистемы трубопровода Manifolds) и двух устройств (CargoTankUnit и LoadDischargeUnit).



а б

Рис. 3. а) Гибридное поведение FluidMap в состоянии Pipe.Fluid_state; б) Система Cargo

2.4.1 Пакет устройств Devices

Как уже упоминалось, классы устройств содержат необходимые экземпляры элементарных классов и реализуют какие-либо функциональные узлы. Например, класс CargoTankUnit состоит из экземпляров таких классов как Tank, Pump, Heater, Pipe, Valve, RetainingValve, ValveRemote и связей между ними, тип которых также описан в библиотеке Math_Hydraulics. Класс LoadDischargeUnit включает экземпляры Pipe, Valve и другие более специальные классы. Классы устройств снабжены внешними переменными для связи с другими устройствами и системами. Кроме того, разработчик имеет возможность переопределять параметры и начальные значения переменных, элементарных составляющих классов.

2.4.2 Пакет подсистем Subsystems

Классы подсистем состоят из ряда экземпляров классов устройств и так же снабжены внешними переменными для связи друг с другом. Подсистема грузовых танков CargoTanks состоит из более чем десяти соединённых друг с другом устройств типа CargoTankUnit. Подсистема Manifolds – из такого же количества устройств LoadDischargeUnit.

2.4.3 Класс Model

Класс Model является основным классом математической модели системы. Во-первых, он объединяет (то есть включает в себя и связывает друг с другом) все её подсистемы (в нашем примере – CargoTanks и Manifolds). Во-вторых, класс Model синхронизует все входящие в него гибридные автоматы. Механизм синхронизации наследуется от базового класса Math_Hydraulics.ModelBase.

Построением класса Model завершается создание математической модели системы. Спроектированная таким образом модель позволяет имитировать поведение достаточно сложных многокомпонентных систем. Полученные модели состоят из сотен экземпляров различных гибридных классов, а в процессе работы в реальном времени

решаются системы, состоящие из тысяч уравнений. В следующем разделе будет описана среда моделирования Model Vision Studium.

3. Средство разработки Model Vision Studium

В качестве средства разработки использовалась специальная версия системы автоматизации моделирования MvStudium 6 [5] (коммерческое наименование Rand Model Designer 6). Эта система поддерживает:

- объектно-ориентированное моделирование в рамках подмножества языка UML,
- физическое моделирование в рамках подмножества языка Modelica,
- определяемые пользователем типы данных и
- автоматически генерирует выполняемый программный код по математическому описанию.

В целом, этот инструмент вполне обеспечивает требуемую технологию разработки.

Наиболее тяжелым оказалось требование работы автоматически сгенерированных встроенных моделей в реальном времени. Для этого должны очень быстро работать численные методы и достаточно быстро (в пределах десятой доли секунды) производиться сборка, анализ и преобразование текущей совокупной системы уравнений (использование расширенных диаграмм состояний UML – карт поведения – для описания гибридного поведения требует проведения этих действий на стадии выполнения). В значительной степени эти требования противоречивы: опыт показал, что для достижения большой скорости численных методов необходимо проведение символьных преобразований для упрощения исходной системы уравнений, а также выявление в ней специальных структур, независимых блоков и т. д., в то же время выполнения такого анализа и преобразований весьма ограничено. В результате удалось найти приемлемый компромисс.

Опыт показал наличие существенных проблем при отладке промышленных моделей большой сложности. Возникают ситуации, когда конкретное соединение стандартных компонентов приводит к некорректной ситуации, не предусмотренной разработчиками библиотеки классов: в определенных точках совокупная система уравнений становится либо структурно некорректной (недоопределенной, переопределенной или вырожденной) или численно вырожденной. Определить, какие именно компоненты «виновны» в данной ситуации для систем большой размерности достаточно сложно. Поэтому средства отладки были дополнены возможностями визуализации матрицы Якоби и ее собственных чисел для текущей системы уравнений, а также структурной матрицы текущей совокупной системы уравнений, которые дают определенную «наводящую» информацию разработчикам модели. Для отработки логики переключений насущной также является возможность интерактивной пошаговой отладки дискретных действий.

Литература

1. Transas Group: Transas Marine Products and Services – http://www.transas.com/products.
2. **Колесов Ю. Б., Сениченков Ю. Б.** Моделирование систем. Динамические и гибридные системы. – С.-Петербург, БХВ, 2006.
3. **Колесов Ю. Б., Сениченков Ю. Б.** Моделирование систем. Объектно-ориентированный подход. – С.-Петербург, БХВ, 2006.
4. Model-based design for control systems with Simulink – Mathworks, Inc., 2005.
5. Model Vision Studium – http://www.mvstudium.com.