УДК 681.3.0

Пинчук В. П.[1], Подковалихина Е. А.[2]

[1]Канд. физ.-мат. наук, доцент Запорожского национального технического университета
[2]Канд. физ.-мат. наук, доцент Запорожского национального технического университета

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД ДЛЯ ПОСТРОЕНИЯ МОДЕЛЕЙ СЛОЖНЫХ СИСТЕМ

Предложена методика построения динамической модели системы с локализованными свойствами на основе объектно-ориентированного подхода. Рассмотрен пример системы автоматического регулирования температуры рабочего тела с пропорционально-интегрально-дифференциальным законом регулирования, приведена методика построения классов и функций, составляющих программную реализацию модели, обсуждаются возможности предлагаемой методики.

Ключевые слова: компьютерное моделирование систем, моделирование динамики системы, объектно-ориентированный подход, система терморегулирования.

ВВЕДЕНИЕ

Параллельно с развитием аппаратных и программных средств вычислительной техники интенсивно эволюционируют и общие подходы моделирования сложных систем различной природы. Однако имеющиеся в настоящее время методы построения динамических моделей технических систем все же не удовлетворяют в полной мере растущим запросам разработчиков систем различной этимологии [1, 2]. Остается открытым вопрос о формализации синтеза компьютерной модели сложной системы из заданного множества субмоделей. Определенным потенциалом в создании более продвинутых методов моделирования систем обладает подход, который называют объектно-ориентированным.

МЕТОДИКА ПОСТРОЕНИЯ ДИНАМИЧЕСКОЙ МОДЕЛИ СИСТЕМЫ

Ниже рассмотрена методика построения динамической модели технической системы на основе объектно-ориентированного подхода и ее программной реализации. Хорошим примером (не слишком громоздким и, в то же время, достаточно характерным) системы как объекта моделирования, является система автоматического регулирования температурой некоторого рабочего тела, структура которой показана на рис. 1. Модель системы в нашем примере состоит из следующих элементов:

1) рабочее тело;
2) нагреватель;
3) регулятор;

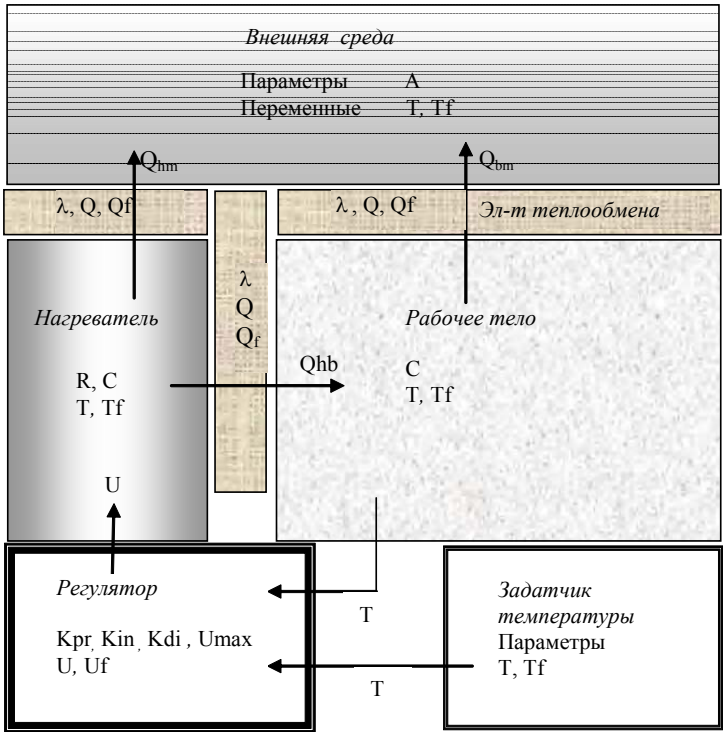

Рис. 1. Схема взаимодействия объектов в системе терморегулирования

4) элемент теплообмена «нагреватель – внешняя среда»;

5) элемент теплообмена «рабочее тело – внешняя среда»;

6) элемент теплообмена «нагреватель – рабочее тело»;

7) задатчик температуры;

8) внешняя среда.

Принимается следующая методология моделирования.

Элементы системы типизированы, т. е. каждый из элементов системы принадлежит определенному виду. Модель элемента системы данного вида представлена соответствующим классом. Состояние элемента системы определяется внутренними процессами, отображаемыми в субмодели, и воздействиями на него других элементов системы. Состояние элемента системы описывается набором переменных его состояния.

Каждый из элементов системы получает информацию о состоянии тех элементов системы, которые оказывают на него влияние. Обобщенная информация о наличии такого влияния для каждого элемента системы представлена матрицей межэлементных воздействий M, которая для рассматриваемой системы имеет следующий вид:

	1	2	3	4	5	6	7	8
1	0	0	0	0	1	1	0	0
2	0	0	1	1	0	1	0	0
3	1	0	0	0	0	0	1	0
4	0	1	0	0	0	0	0	1
5	1	0	0	0	0	0	0	1
6	1	1	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

Номера строк и номера столбцов в этой матрице соответствуют номерам элементов системы, перечисленным ранее. Единичное значение элемента матрицы M_{ik} означает, что элемент с номером i испытывает воздействие со стороны k-того элемента системы. Матрица M используется для построения интерфейсов функций, описывающих поведение элементов системы.

Моделирование поведения системы выполняется на заданном отрезке времени $[0, t_m]$. Временной шаг моделирования может быть переменным. При этом выполнение каждого шага включает в себя 3 этапа.

1. Каждый элемент системы получает информацию о состояниях воздействующих на него в данный момент времени элементов.

2. Каждый из элементов системы вычисляет свое состояние в следующий момент времени с учетом величины текущего шага моделирования (не переходя при этом в следующее состояние). Вычисления выполняются на основе соответствующей субмодели элемента системы.

3. Каждый из элементов системы реализует (переводит себя в) следующее состояние.

В соответствии с идеологией объектно-ориентированного моделирования (ООМ) все элементы модели системы типизированы, т. е. относятся к определенному классу и каждый элемент модели системы представляется объектом соответствующего класса. При этом структура, свойства и поведение объекта данного класса, порождаемого моделирующей программой, однозначно определяется описанием этого класса. Класс определяет информационную структуру элемента модели системы и содержит набор функций (методов), определяющих эволюцию его состояния. При этом структура межэлементных взаимодействий, определяемая матрицей M, представлена в соответствующих классах в виде списков аргументов функций – членов класса, осуществляющих выполнение второго этапа очередного шага моделирования системы.

СТРУКТУРА И ОПИСАНИЕ КЛАССОВ ЭЛЕМЕНТОВ СИСТЕМЫ

Обобщенную структуру класса, описывающего произвольный элемент системы, можно представить следующим образом:

```
class <имя типа элемента>
{ <список параметров модели элемента системы
с указанием их типов>
< описание переменных состояния текущего мо-
мента времени >
< описание переменных состояния следующего
момента времени >
void next(список аргументов – состояний воз-
действующих элементов);
void step();                    // реализация
следующего состояния системы
< конструкторы и другие стандартные элементы
класса>
};
```

Приведем описание каждого из классов рассматриваемой модели. Описание конструкторов и других стандартных элементов классов опустим для краткости.

Класс Body (рабочее тело)

```
class Body
{public:
double C;
double T, Tf;
void        next(Inter<Heater,Body>&,
Inter<Body,Medium>&);
void step() { T=Tf; }
Body() { C=0.0; T=0.0; Tf=0.0; }
Body(double c, double x) { C=c; T=x; }
};
void Body::next(Inter<Heater,Body>& hb,
Inter<Body,Medium>& bm)
{ Tf= T+(h/C)*(hb.Q-bm.Q);
}
```

Модель элемента системы типа Body имеет один параметр: теплоемкость рабочего тела C, по одной переменной состояния для текущего и следующего моментов времени T и Tf (температура рабочего тела). Функция next, вычисляющая следующее состояние системы (т. е. значение Tf), через список своих аргументов получает информацию о состояниях элементов теплообмена «нагреватель – рабочее тело» и «рабочее тело – внешняя среда». Приведенный алгоритм для функции next

класса Body соответствует рабочей формуле первого порядка интегрирования дифференциального уравнения теплового баланса рабочего тела:

$$\frac{dT}{dt} = \frac{1}{C}(Q_{hb} - Q_{bm}), \quad (1)$$

где Q_{hb} – тепловой поток от нагревателя к рабочему телу, Q_{bm} – тепловой поток от рабочего тела во внешнюю среду (в приведенном описании класса эти величины имеют имена hb.Q и bm.Q).

Класс Heater (нагреватель)

```
class Heater
{ public:
double R, C;
double T, Tf;
void next(Regul&, Inter<Heater,Medium>&,
Inter<Heater,Body>&);
void step() { T=Tf; }
Heater(double r, double c, double To) { R=r;
C=c; T=To; }
};
void Heater::next(Regul& reg,
Inter<Heater,Medium>& hm, Inter<Heater,Body>&
hb)
{ double P=sqr(reg.U)/R;
Tf=T+(h/C)*(Phm.Q-hb.Q);
}
```

Модель нагревателя имеет два параметра: R – электрическое сопротивление нагревателя, C – его теплоемкость. Параметром состояния является температура (переменные T и Tf). Функция next получает информацию о состоянии регулятора, а также элементов теплообмена «нагреватель – внешняя среда» и «нагреватель – рабочее тело». Алгоритм функции next включает в себя вычисление тепловой мощности нагревателя (входного теплового потока):

$$P = \frac{U^2}{R}, \quad (2)$$

и температуры для следующего момента времени. Для вычисления последней использована рабочая формула для уравнения теплового баланса нагревателя такого же вида, как и в случае рабочего тела.

Класс Regul (регулятор)

```
class Regul
{public:
double Umax, Kpr, Kin, Kdi;
double Uf, U;
void next(Body&, Targ&);
void step() { U=Uf; }
Regul(double Um, double kpr, double kin, double
kdi)
{Umax=Um; Kpr=kpr; Kin=kin; Kdi=kdi; U=0; }
};
void Regul::next(Body& body, Targ& targ)
{static int M; M++;
static double S=0.0;
static Body body0;
static Targ targ0;
if (M==1) { body0=body; targ0=targ; }
double Dx=targ.T-body.T, Dxo=targ0.T-body0.T,
D=(Dx-Dxo)/h;
S+=h*(Dx+Dxo)/2;
Uf= Kpr*Dx+Kin*S+Kdi*D;
if (Uf<0) Uf=0.0;
if (Uf>Umax) Uf=Umax;
body0=body; targ0=targ;
}
```

Субмодель регулятора соответствует универсальному пропорционально-интегрально-дифференциальному закону регулирования. Параметрами модели регулятора являются величины: Kpr – коэффициент пропорционального регулирования, Kin – коэффициент интегрального регулирования, Kdi – коэффициент дифференциального регулирования,Umax – максимальное напряжение на выходе регулятора. Параметром состояния является напряжение на выходе регулятора (переменные U, Uf).

Функция next построена на основе уравнения пропорционально-интегрально-дифференциального регулирования. Напряжение U, выдаваемое регулятором, определяется значением и поведением во времени температуры рабочего тела T и температуры задатчика Z:

$$U = K_{pr}(Z-T) + K_{in}\int_0^t (Z-T)dt + K_{di}\frac{d}{dt}(Z-T), \quad (3)$$

Класс Inter (элемент теплообмена)

Основным назначением субмодели вида «элемент теплообмена» является описание механизма передачи тепла от одного элемента системы к другому. Хотя такая субмодель может учитывать и другие сущности. В нашем примере элементы теплообмена являются безинерционными и, таким образом, представляют механизм теплопередачи в «чистом виде».

При построении данного класса используются возможности параметризованных классов (шаблонов). Класс Inter является шаблоном с параметрами type1 и type2, с помощью которых передаются типы воздействующих объектов – элементов системы.

```
template <class type1, class type2>
class Inter
{ public:
double Ld;
double Q, Qf;
Inter(double ld) { Ld=ld; Q=0; }
void next(type1,type2);
void step() { Q=Qf; }
};
template <class type1, class type2>
void Inter<type1,type2>::next(type1 A, type2 B)
{ Qf=Ld*(A.T-B.T);
}
```

Объект класса Inter содержит один параметр (Ld – коэффициент теплопроводности соответствующего материала) и один параметр состояния (Q, Qf – передаваемые тепловые потоки). Функция next вычисляет тепловой поток в соответствии с уравнением:

$$Q = \lambda(T_a - T_b), \quad (4)$$

где T_a, T_b – температуры источника и приемника тепла, λ – коэффициент теплообмена.

Класс Targ (задатчик температуры)

Задатчик температуры является целеустанавливающим элементом системы. Он представляет закон изменения температуры со временем или просто заданную температуру. Состояние задатчика в нашем примере не зависит от состояний других элементов системы.

```
class Targ
{public:
double T, Tf;
Targ() { T=Tf=Tmin; }
Targ(double A) { T=A; }
void next();
void step() { T=Tf; }
};
void Targ::next()
{const float A=300;
if (t<tm/2) Tf=t/tm*A; else Tf=A*(1.0-t/tm);
}
```

Параметров данная субмодель в нашем примере не имеет. Вместо этого, параметры задатчика могут быть инкапсулированы в функцию – член класса next. В нашем примере мы имеем один такой параметр. Параметром состояния задатчика является выдаваемая им температура (переменные T, Tf).

Функция next содержит принятый закон изменения температуры: линейный рост (первая фаза) и линейное снижение (вторая фаза) температуры рабочего тела:

$$T = \begin{cases} T_f = \dfrac{t}{t_m} A, \text{ если } t < \frac{1}{2} t_m, \\ T_f = A(1 - \dfrac{t}{t_m}), \text{ если } t \geq \frac{1}{2} t_m. \end{cases} \quad (5)$$

Параметры A – максимальная температура процесса, t_m – общее время моделирования системы удобно представить как общие параметры моделирования, присутствующие в программе в виде глобальных переменных.

Класс Medium (внешняя среда)

Данный класс представляет субмодель внешней среды.

```
class Medium
{public:
double A1, A2;
double T, Tf;
Medium(double a1, double a2){ A1=a1; A2=a2; T=A1; }
void next();
void step() { T=Tf; }
};
void Medium::next()
{if (t<tm/2) Tf=A1; else Tf=A2;
}
```

Субмодель имеет два параметра (A1 и A2) и одну переменную состояния – температура внешней среды (T, Tf).

Функция next содержит информацию об изменении состояния внешней среды: ее температура в момент времени $t = t_m/2$ скачкообразно изменяется от значения A1 до A2.

РЕЗУЛЬТАТЫ МОДЕЛИРОВАНИЯ

Результаты моделирования приведены на рис. 2 и 3. На рисунках приведены следующие кривые: температура задатчика, напряжение на выходе регулятора, температура нагревателя, температура рабочего тела, все величины как функции времени. Результаты моделирования на рис. 2 соответствуют следующим условиям: температура внешней среды постоянна, задатчик формирует сигнал в форме П-импульса. Параметры модели:

1. Задатчик: Tmin=10, Tmax=100.
2. Внешняя среда: Tmedi=10.
3. Элементы теплообмена: Lbm=10, Lhm=0.1, Lhb=15.
4. Рабочее тело: Cbody=20.
5. Нагреватель: Rheat=100, Cheat=2.
6. Регулятор: Umax=400, Kpr=20, Kin=3, Kdi=0.1.

Рис. 3 соответствует таким условиям: температура внешней среды постоянна, задатчик формирует сигнал в форме треугольного импульса. Параметры модели:

1. Задатчик: Tmin=10, Tmax=100.
2. Внешняя среда: Tmedi=0.
3. Элементы теплообмена: Lbm=10, Lhm=0.1, Lhb=15.
4. Рабочее тело: Cbody=20.
5. Нагреватель: Rheat=100, Cheat=2.
6. Регулятор: Umax=400, Kpr=20, Kin=12, Kdi=0.

ВЫВОДЫ

На примере конкретной системы показана результативность объектно-ориентированного подхода для построения компьютерной модели, позволяющей отслеживать фазовую траекторию системы. Показаны особенности построения программной реализации модели на основе объектно-ориентированного подхода. Основными преимуществами такого подхода являются следующие.

1. Модульность, позволяющая с минимальными затратами вносить функциональные и структурные изменения в модель анализируемой системы.

2. Возможность формализации процесса построения программной реализации динамической модели системы с локализованными свойствами различной этимологии и природы.

3. Потенциальная возможность построения моделей систем высокого уровня сложности, включающих большое число элементов и связей.

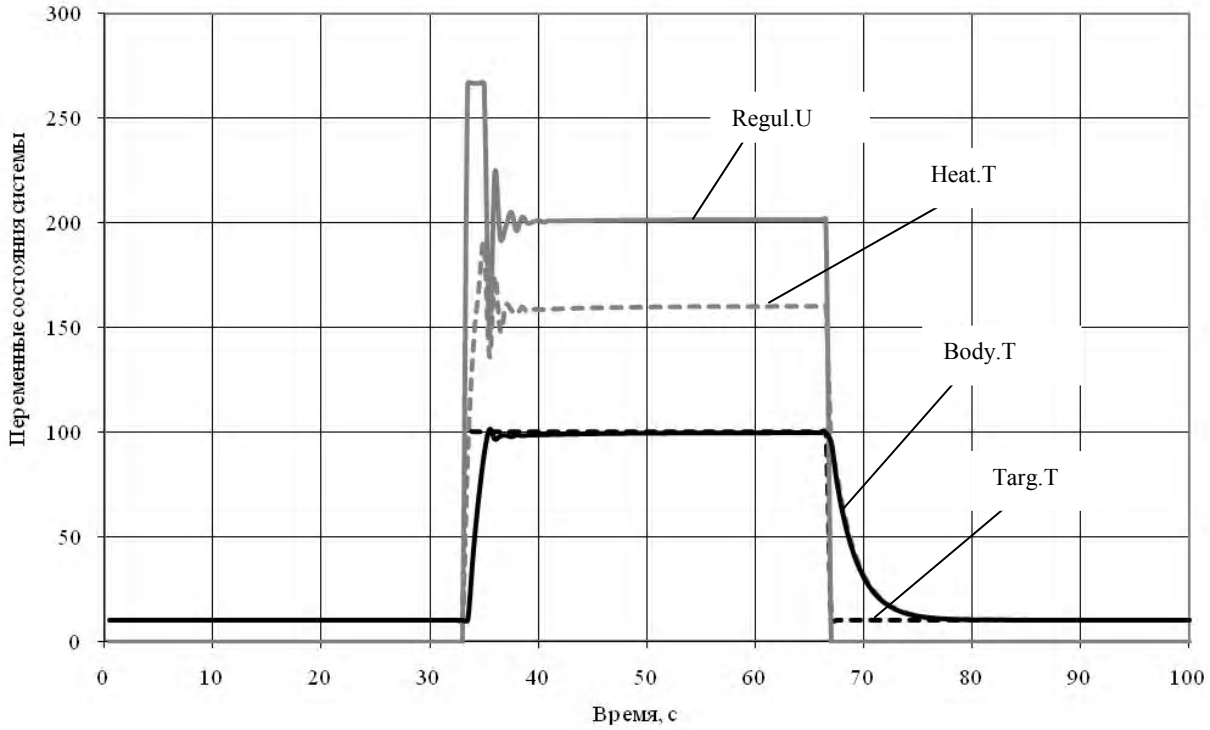

Рис. 2. Результат моделирования при ступенчатом изменении задаваемой температуры

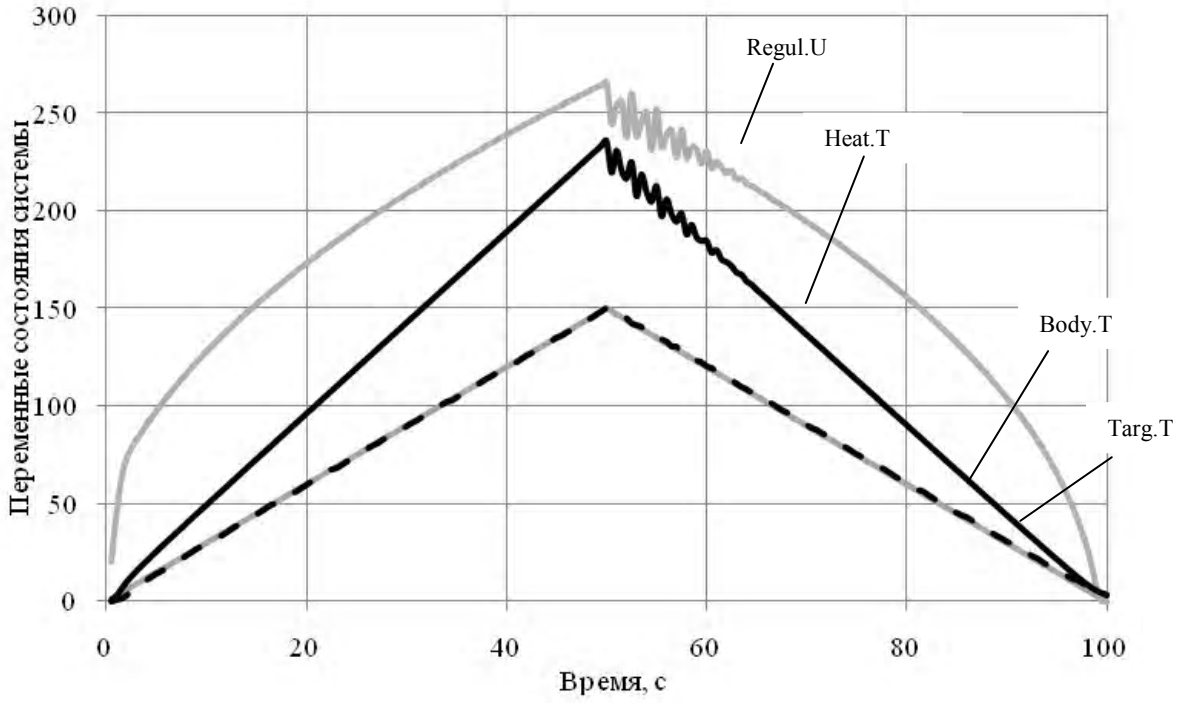

Рис. 3. Результат моделирования для функции задатчика треугольной формы

СПИСОК ЛИТЕРАТУРЫ

1. *Колесов, Ю. Б.* Объектно-ориентированное моделирование сложных динамических систем / Ю. Б. Колесов. – СПб. : Изд-во С.Пб. ГПУ, 2004. – 239 с.

2. *Колесов, Ю. Б.* Моделирование систем. Динамические и гибридные системы / Ю. Б. Колесов, Ю. Б. Сениченков. – С. Пб. : БХВ-Петербург, 2006. – 224 с.

Стаття надійшла до редакції 15.03.2011.

Пінчук В. П., Подковаліхіна О. О.

ОБ'ЄКТНО-ОРІЄНТОВАНИЙ ПІДХІД ДЛЯ ПОБУДОВИ МОДЕЛЕЙ СКЛАДНИХ СИСТЕМ

Запропоновано методика побудови динамічної моделі системи з локалізованими властивостями на основі об'єктно-орієнтованого підходу. Розглянуто приклад системи автоматичного регулювання температури робочого тіла з пропорціонально-інтегрально-диференціальним законом регулювання, наведено методику побудови класів і функцій, що складають програмну реалізацію моделі, обговорюються можливості запропонованої методики.

Ключові слова: комп'ютерне моделювання систем, моделювання динаміки системи, об'єктно-орієнтований підхід, система терморегулювання.

Pinchuk V. P., Podkovalihina E. A.

OBJECT-ORIENTED APPROACH FOR MODELS DESIGNING OF THE COMPLEX SYSTEMS

A method for constructing a dynamic model of the system with localized features based on object-oriented approach was proposed. An example of a system for automatic temperature control of the working body with a proportional-integral-differential law of regulation was considered.

A technique for constructing classes and their functions that comprise the software implementation of the model was leaded, the opportunities of the proposed method were discussed.

Key words: computer simulation of the systems, modeling the dynamics of the system, object-oriented approach, the thermal control system.