СРЕДА ИМИТАЦИОННОГО СОБЫТИЙНОГО МОДЕЛИРОВАНИЯ

Е. А. Бабкин, В. В. Разиньков (Курск)

В настоящее время существуют и продолжают развиваться различные виды программных средств имитационного моделирования, используемых как для решения сложных прикладных задач, так и для обучения. Выделяются следующие разновидности инструментария имитационного моделирования [1, 2]: библиотеки поддержки имитационного моделирования, языки имитационного моделирования, интегрированные среды. Библиотеки поддержки предоставляют низкоуровневый интерфейс разработки программной модели, связанный с некоторым языком программирования, что усложняет процесс разработки (особенно для сложных моделей), однако они обладают наибольшей гибкостью для разработчика. Языки имитационного моделирования (GPSS и др.) освобождают от знания какого-либо языка программирования, предоставляя в качестве инструментария разработки набор собственных операторов определенного формата. Интегрированные среды являются наивысшем развитием программных средств имитационного моделирования и, помимо визуального процесса разработки и выполнения модели, предоставляют широкие возможности статистического анализа результатов моделирования.

Одной из проблем при обучении основам моделированию, как показывает опыт, является малая эффективность обучения, как с использованием аналитических моделей, так и моделей на языках имитационного моделирования, поскольку построение аналитических моделей возможно только для простых систем, а моделирование на языках имитационного моделирования, позволяющих описывать сложные системы, достаточно трудоемко. Кроме того, интегрированные среды моделирования, использующие визуальное представление моделей, достаточно громоздки, имеют большую стоимость и ориентированы на промышленное применение, а не на учебный процесс. Также данные среды не позволяют изучать механизм моделирования, представления моделей в терминах событий. Они дают возможность изучать технологию моделирования на уровне внешнего интерфейса среды имитационного моделирования. Последний фактор имеет большое значение, поскольку при изучении компьютерного моделирования ставятся различные задачи: от изучения внутренней организации систем моделирования до решения задач с помощью этих систем. В связи с этим актуальной является разработка среды имитационного моделирования, ориентированной на процесс обучения, с графическим вводом и выводом, что значительно снижает трудоемкость разработки, ввода и проверки модели, повышает наглядность процесса моделирования, позволяет проводить различные эксперименты с моделью, по ходу эксперимента изменять его условия и наблюдать за состоянием всех объектов и самой системы моделирования. Все это повышает эффективность обучения моделированию.

Рассматриваемая среда имитационного событийного моделирования дискретных систем ESimPL базируется на использовании событийных графов [3, 4] для описания процесса функционирования исследуемой системы.

Событийные модели представляют процесс функционирования исследуемой системы в терминах событий [5]. Выделяются два уровня представления событийных моделей: уровень математической модели – событийный граф, и уровень программной модели – в виде совокупности процедур макрособытий на языке программирования высокого уровня с использованием библиотеки поддержки имитационного моделирования, представляющей собой совокупность процедур и функций или классов на языке программирования модели. В связи с этим событийная система моделирования дис-

кретных систем включает два языка: язык визуального моделирования и язык программирования модели.

Для визуального представления событийных моделей в соответствии с подходом, изложенным в работе [5], были выделены графические и внутренние элементы модели.

Графические элементы составляют основу языка визуального событийного моделирования. Элементами модели, имеющими свои графические обозначения являются событие, условие, начало и окончание распараллеливания процесса и дуги четырех типов: мгновенного следования, следования с задержкой, мгновенной отмены и отмены события с задержкой [5]. Основной графический элемент модели – событие.

Внутренние элементы, не представляемые графически, – это статические и динамические объекты и переменные модели. Статическими объектами являются средства (каналы), очереди и ресурсы, динамическими объектами – процессы и активности. Каждому объекту ставится в соответствие два события изменения состояния: занятие и освобождение для статического объекта, начало и конец выполнения для динамического. Переменные разделяются на две разновидности: глобальные переменные модели и переменные процесса.

Языком программирования моделей среды ESimPL является язык высокого уровня C++, дополненный библиотекой поддержки имитационного моделирования. Язык моделирования ESimPL является развитием событийно-ориентированного языка SMPL [6] и разработан с использованием объектно-ориентированного подхода. Программная модель генерируется на языке C++, в тексте которой используются операторы языка ESimPL, реализованные в виде вызова методов граничного класса библиотеки поддержки имитационного моделирования. Язык моделирования определяется набором операторов, используемых при разработке программной модели.

Основу библиотеки составляют 7 классов, разделенных на 3 группы:

– граничный (интерфейсный) класс, реализующий функциональность языка моделирования, открытые методы данного класса определяют полный набор операторов языка; класс содержит в себе: списки регистрируемых в модели объектов (средств, очередей, ресурсов и процессов), переменную модельного времени, переменную текущего процесса, список запланированный и прошедших событий и многие другие параметры;

– классы, реализующие поведение статических и динамических объектов модели, накапливающие необходимые статистические данные эксперимента (результаты моделирования);

– вспомогательные классы, поля которых хранят информацию о постановке процессов в очереди и занятии ресурсов во время работы программной модели.

В состав операторов языка моделирования, реализованных в библиотеке поддержки имитационного моделирования, входят следующие группы операторов:

– операторы определения в модели статических объектов (средств, очередей, ресурсов);

– операторы изменения и проверки состояний статических объектов;

– операторы создания и удаления в модели динамических объектов (процессов);

– операторы работы с событиями (планирование, отмена и выбор очередного события из списка запланированных);

– операторы генерации псевдослучайных величин с вероятностными распределениями определенного вида (равномерное, экспоненциальное, треугольное);

– оператор генерирования отчёта по результатам моделирования.

Программные средства системы ESimPL включают CASE-средство разработки событийного графа дискретной системы, систему поддержки имитационного моделирования в виде библиотеки классов и шаблоны примеров с детальными пояснениями.

В CASE-средстве реализован оконный интерфейс для ввода, редактирования и удаления внутренних элементов модели (рис. 1). При вводе статических и динамических объектов среда автоматически генерирует события изменения состояний этих объектов, занося их в общий список. Данный список в дальнейшем используется для построения событийного графа. Введенные переменные модели используются для параметризации элементов графа на этапе его построения.

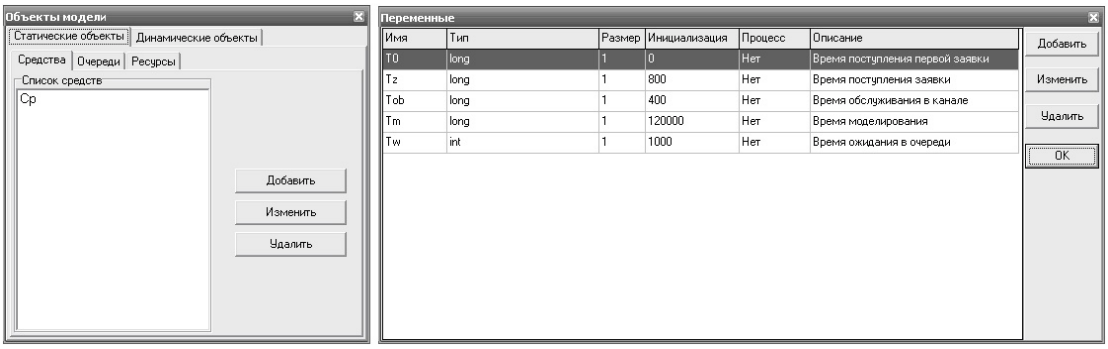

Рис. 1. Окна регистрации в модели внутренних элементов

Главное окно CASE-средства (рис. 2) предоставляет интерфейс разработчика, позволяющий в интерактивном режиме вводить и редактировать имитационный событийный граф. Кнопки боковой инструментальной панели соответствуют графическим элементам событийного графа и предназначены для выбора режима работы в рабочей области и ввода вершин и дуг событийного графа. В главном меню программного средства реализованы все необходимые операции над событийном графом. В число таких операций входит:

– проверка разрабатываемого событийного графа функционирования системы на соответствие правилам построения событийных графов и вывод ошибок;

– преобразование исходного событийного графа в программно-реализуемую форму (выделение макрособытий);

– проверка корректности программно-реализуемой формы графа с выделенными макрособытиями;

– настройка внешнего вида рабочей области и элементов событийного графа;

– сохранение разработанной модели в текстовый файл, содержащий описание модели на языке разметки XML, а так же загрузка модели из него;

– генерирование кода программной модели на языке программирования C++.

Проверка графа на соответствие правилам построения событийных графов осуществляется на различных этапах разработки модели: в процессе ввода вершин и дуг графа, после создания исходного событийного графа, после преобразования исходного событийного графа в программно-реализуемую форму графа с выделенными макрособытиями.

Формат XML, в котором сохраняются модели, является открытым и предоставляет возможность в текстовом редакторе просмотреть параметры сохраненной модели и производить их коррекцию.

Сгенерированная модель (рис. 3) может быть откорректирована разработчиком и сохранена в файле. Сохраненная модель в дальнейшем компилируется в среде программирования Borland C++ Builder. Скомпилированная модель позволяет выполнять прогон в режимах калибровки модели: режиме с выводом состояний всех статических объектов системы и режиме трассировки макрособытий и в режиме компьютерного эксперимента, в котором выводится только отчет о моделировании.

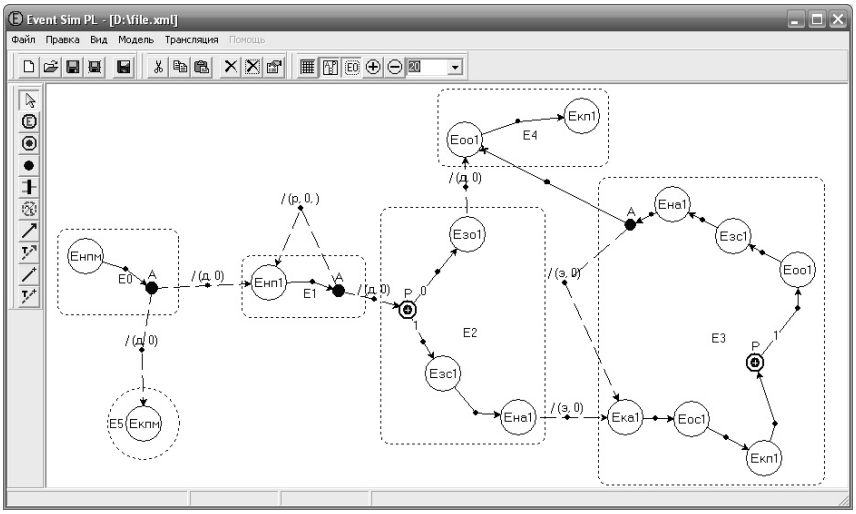

Рис. 2. Главное окно CASE-средства среды ESimPL

Использование среды ESimPL и ее библиотеки позволит:

– визуализировать а, следовательно, и сократить трудоемкость и время, процесса разработки событийных моделей;

– обучать описанию процессов поведения систем в терминах событий;

– сохранить доступность модели на программном уровне, что позволяет ее дорабатывать, изменять и, главное, обучать соответствию программной модели и событийной модели;

– программировать модель непосредственно на языке C++ с использованием библиотеки и выполнять прогоны;

– встраивать сгенерированную модель как элемент в другие программные продукты.

Программная модель

Файл Правка

```
#include <iostream>
#include "ClassImit.h"
class TUserProcess :public TProcess {
public:
__fastcall TUserProcess(void);   // конструктор класса процесса
};
__fastcall TUserProcess::TUserProcess(void) :TProcess()
{ ; }
// переменные пользователя
long T0=0;   // Время поступления первой заявки
long Tz=800;   // Время поступления заявки
long Tob=400;   // Время обслуживания в канале
long Tm=120000;   // Время моделирования
int Tw=1000;   // Время ожидания в очереди

typedef TModel<TUserProcess> TUserModel;   // Тип класса модели
TUserModel * mymodel;   // Переменная класса модели
TUserProcess * current_process;   // Переменная текущего процесса
int eventNumber;   // Переменная текущего макрособытия

// Переменные статических объектов
Equip equip1;    // переменная средства 'Ср'
Queue queue1;    // переменная очереди 'Оч'

void E0(void)
{
mymodel = new TUserModel();
mymodel->printHeader();
mymodel->printAndSetupOptions();

equip1=mymodel->Create_equip("Ср");   // регистрация в модели средства 'Ср'
queue1=mymodel->Create_queue("Оч",0,0);   // регистрация в модели очереди 'Оч'

mymodel->schedl(1,0,mymodel->Main_process);
mymodel->schedl(5,0,mymodel->Main_process);
}
```

Рис. 3. Сгенерированная программная модель

Достоинствами среды ESimPL являются наглядное визуальное проектирование событийного алгоритма функционирования системы; использование для разработки программной модели объектно-ориентированного подхода; верификация разработанной модели на основе выдаваемых системой сообщений об исключительных ситуациях, возникающих в процессе моделирования; удобство представления результатов моделирования; небольшое время выполнения прогона программной модели. Внедрение и опыт эксплуатации среды в учебном практикуме по дисциплине «Компьютерное моделирование» для различных специальностей в вузе подтвердило ее эффективность.

Литература

1. **Кельтон В., Лоу А.** Имитационное моделирование. Классика. СПб.: Питер, 2004. 847 с.
2. **Савина О. А., Погорелов А. С.** Разработка высокоуровневой среды имитационного моделирования JaSim. // III Всероссийская научно-практическая конференция ИММОД-2007 / Сб. докладов. Т. I. СПб.: ЦНИИТС, 2007. С. 302 – 305.
3. **Schruben L. W.** Simulation Modeling with Event Graphs. // Communications of the ACM. 1983. Vol. 26. Num. 11. P. 957–963.
4. **Бабкин Е. А.** Событийные модели дискретных систем / Курск. гос. ун-т. Курск, 2005. 18 с. Деп. в ВИНИТИ 14.01.05, № 30–В2005.
5. **Бабкин Е. А.** О синтезе событийных моделей дискретных систем // Ученые записки. Электронный научный журнал Курск. гос. ун-та, Эл № 77-26463, № 1, 2006. http://www.scientific-notes.ru/pdf/s15.pdf.
6. Автоматизация проектирования вычислительных систем. Языки, моделирование и базы данных / Под ред. М. Брейера. М.: Мир, 1979.