УДК 681.324

А.В. СКАТКОВ, Д.Ю. ВОРОНИН

Севастопольский национальный технический университет, Украина

ОБЕСПЕЧЕНИЕ ГАРАНТОСПОСОБНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ОСНОВЕ АВТОМАТИЗАЦИИ СОПРЯЖЕНИЯ РАЗНОРОДНЫХ ПРОГРАММНЫХ СРЕД

Рассмотрена задача обеспечения гарантоспособности программного обеспечения. Приводится подход, способный минимизировать дефекты взаимодействия за счет автоматизации сопряжения разнородных программных сред. В работе использован механизм операционной системы, служащий для обеспечения гарантоспособного взаимодействия между приложениями (передача сообщений). В качестве примера рассматривается организация взаимодействия высокоуровневой среды разработки приложений Delphi со специализированной программной средой имитационного моделирования GPSSW. При учете многократной модификации параметров модели и большого объема получаемых в результате многократных прогонов модели данных, разработанная система помогает избавиться от большого числа ошибок, вызванных участием человека. Предложенный подход планируется внедрить в систему поддержки принятия решения по управлению ресурсами распределенной вычислительной системы.

гарантоспособность, программное обеспечение, дефекты взаимодействия, разнородные программные среды

Введение

Практика показала, что применение классической теории надежности к программным средствам не всегда достаточно эффективно. В [1] приводятся возможные причины такого положения вещей, и вводится понятие гарантоспособности. Там же поясняются базовые понятия и таксономии гарантоспособности. Известно, что гарантоспособные системы обладают рядом свойств, соблюдение которых исключает существенные материальные убытки и катастрофы различного масштаба [1].

Дефекты взаимодействия являются одним из трех возможных типов дефектов [2] и являются либо следствием воздействия информационных атак, либо ошибок персонала [1].

Целью статьи является рассмотрение и решение ряда вопросов, связанных с обеспечением гарантоспособности программного обеспечения на основе автоматизации разнородных программных сред.

Для начала обоснуем необходимость сопряжения разнородных программных средств при имитационном моделировании и решении оптимизационных задач. В таблице 1 приведена сравнительная характеристика различных программных комплексов, используемых при имитационном моделировании сложных систем.

Анализировались программные средства: Delphi, C Builder, GPSS World, AnyLogic, сопряженная система DG.

В качестве основных критериев оценки эффективности программных комплексов, используемых при имитационном моделировании использовались:

– гибкость языковых конструкций;

– лаконичность;

– универсальность;

– использование готовых библиотек алгоритмов на Паскале и Си;

– создание интерфейса и анимация процесса моделирования;

– использование библиотек моделей написанных на GPSS;

– экспорт и импорт данных из различных приложений;

– расширенные возможности по организации и проведению экспериментов с моделью;

– объектно-ориентированный подход к созданию моделей.

Таблица 1

Сравнительная характеристика программных комплексов для имитационного моделирования

Название системы / Характеристики	Delphi	C Builder	GPSS World	AnyLogic	Сопряженная система DG
Гибкость языковых конструкций	+	+	–	+	+
Лаконичность	–	–	+	+	+
Универсальность	+	+	–	+	+
Использование готовых библиотек алгоритмов на Паскале и Си	+	+	–	–	+
Создание интерфейса и анимация процесса моделирования	+	+	–	+	+
Использование библиотек моделей написанных на GPSS	–	–	+	–	+
Экспорт и импорт данных из различных приложений	+	+	–	+	+
Расширенные возможности по организации и проведению экспериментов с моделью	–	–	–	+	+
Объектноориентированный подход к созданию моделей	+	+	–	+	+

По данным табл. 1 видно, что сопряженная среда обладает рядом преимуществ даже по сравнению с наиболее гибкой и современной на сегодняшний день средой имитационного моделирования AnyLogic. Таким образом, разработка такой сопряженной системы является крайне важной и актуальной задачей.

Постановка задачи. Необходимо разработать подход, способный минимизировать дефекты взаимодействия за счет автоматизации сопряжения разнородных программных сред.

Имеются две или более разнородные программные среды. Необходимо автоматизировать процесс обмена данными между ними, для того чтобы минимизировать влияние психофизиологических качеств оператора на гарантоспособность программного обеспечения. В процессе решения поставленной задачи необходимо применять, по возможности, средства, встроенные в операционную систему. Продемонстрировать разработанной подход на примере взаимодействия высокоуровневой среды программирования Borland Delphi 7 со специализированной средой имитационного моделирования GPSS World.

Описание метода

Для сопряжения разнородных программных систем исходя из теории «Операционных систем» известно несколько возможных способов организации взаимодействия. Опишем наиболее простой для понимания метод, основанный на передаче сообщений между приложениями. Рассмотрим общую идею взаимодействия программы, написанной на языке высокого уровня, (далее просто «программы») со средой моделирования GPSS World.

С помощью программы подготавливаются исходные данные для имитационной модели, затем средствами программы вносятся изменения в файл модели. Когда файл с моделью подготовлен, запускается среда моделирования GPSS World, в которую загружается подготовленный файл с моделью и запускается на выполнение. По окончании моделирования, из файла с отчетом в программу импортируются необходимые данные, и среда программирования за ненадобностью закрывается. На рис. 1 изображены основные этапы взаимодействия программы со средой GPSS World.

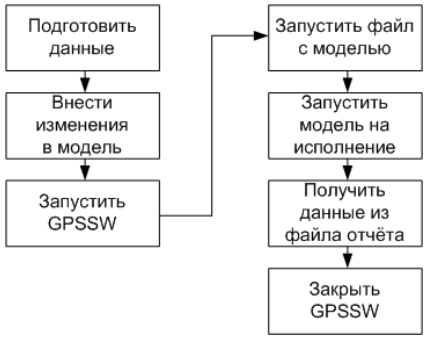

Рис. 1. Основные этапы сопряжения Delphi и GPSS

Рассмотрим эти этапы более детально. Подготовка данных подразумевает формирование списка переменных, которые представляют собой параметры имитационной модели, а также их конкретных значений для данного запуска модели.

Внесение изменений в модель обозначает изменение текстового файла, содержащего имитационную модель, т.е. замену значений переменных в файле с моделью на значения, предназначенные для текущего запуска модели. Целесообразно сначала сделать копию текстового файла с моделью, и дальнейшие изменения производить не с оригиналом, а с копией. Запуск среды моделирования GPSSW представляет собой создание процесса на базе исполнимого файла среды моделирования GPSSW. При запуске исполнимого файла необходимо получить дескриптор процесса и заголовок созданного окна для того, чтобы в дальнейшем знать идентификатор получателя для отправки сообщений. Загрузка модели в GPSS начинается с отправки сообщения в окно GPSSW, предназначенного для вызова диалогового окна открытия файла. Затем необходимо получить идентификатор открывшегося диалогового окна. С помощью полученного идентификатора производится отправка двух сообщений диалоговому окну. Первое – предназначено для передачи имени файла с измененной моделью диалоговому окну, второе – дает указание на открытие этого файла. Запуск модели производится передачей сообщения окну GPSSW, содержащего указание на компиляцию и запуск модели. Момент окончания моделирования определяется по появлению окна в окне GPSSW, содержащего слово «REPORT». Получение данных из отсчета производится в два этапа. На первом этапе по аналогии с загрузкой модели, производится сохранение отчета в текстовый файл, который на втором этапе считывается в программу. Завершение работы среды GPSSW осуществляется уничтожением дескриптора процесса.

Численное решение. В качестве примера реализации описанного подхода рассмотрим задачу об оптимизации процесса конвейерной обработки изделий. Пусть имеется конвейерная лента, на которую поступают детали на обработку с интенсивностью λ. Обработка ведется на двух взаимонезависимых устройствах, для которых заданы времена обработки одного изделия (равны μ_1 и μ_2). Система относится к классу СМО с потерями, так как если оба устройства заняты, а на обработку поступает очередная заявка, то она теряется. Необходимо найти такую интенсивность поступления изделий на конвейер, при которой устройства обработки будут достаточно загружены ($\rho>>0$), а потери заявок будут минимальны.

Целевая функция может быть представлена:

$$\begin{gathered} F(c,\rho_1,\rho_2)=\alpha W_1-\beta W_2\to\max; \\ W_1(\lambda,\mu_1,\mu_2)=\rho_1(\lambda,\mu_1)+\rho_2(\lambda,\mu_2), \\ W_2=\sum c \end{gathered} \qquad (1)$$

где c – потери заявок;

ρ_1,ρ_2 – коэффициенты загрузок устройств;

λ – интенсивность потока изделий;

α, β – коэффициенты важности (компромисс между загрузкой оборудования и потерей заявок).

В рамках примера необходимо построить график зависимости загрузки приборов и количества потерянных заявок от интенсивности поступления деталей. Интенсивность будем вычислять внутри заданного пользователем интервала.

Описательно алгоритм поисковой оптимизации состоит из следующих пунктов:

1. Пока не превышены заданные границы интервала;

1.1. увеличить счетчик на шаг изменения интенсивности;

1.2. модифицировать параметры модели;

1.3. осуществить прогон модели;

1.4. проанализировать результаты прогона (файл отсчета после моделирования);

1.5. вернуться к пункту 1.

2. Построить график зависимости загрузки приборов и количества потерянных заявок от интенсивности поступления деталей.

3. Вывести полученные результаты в виде таблицы.

Очевидно, что данная задача не может быть эффективно решена только средствами среды GPSSW [3], в связи с тем, что для изменения интенсивности потока заявок необходимо останавливать, модифицировать и перезапускать модель. При выполнении этих процедур многократно вручную значительно увеличивается время проведения экспериментов с моделью. В связи с этим целесообразно использовать предлагаемый подход.

Сначала необходимо разработать модель в нотации GPSSW с учетом особенностей взаимодействия среды Delphi [4] с системой GPSSW. Ниже приведена возможная реализация такой модели:

```
Lost VARIABLE 0
Lambda VARIABLE 2

  GENERATE V$Lambda
  GATE NU FAC1,ONE
  SEIZE FAC1
  ADVANCE 23
  RELEASE FAC1
  TERMINATE
ONE    GATE NU FAC2,TWO
  SEIZE FAC2
  ADVANCE 17
  RELEASE FAC2
  TERMINATE
TWO    SAVEVALUE LOST+,1
  TERMINATE

  GENERATE 1200
  SAVEVALUE LOST+,1
  TERMINATE 1
  START 1
```

В виде переменных задаются величины, которые будут передаваться из Delphi, и модифицироваться в процессе построения зависимости. Второй сегмент кода модели представляет собой описание действий с заявкой от её генерации до уничтожения. В третьем сегменте при помощи переменной задается время моделирования.

Перейдем к разработке приложения в среде программирования Delphi. Во-первых, для передачи данных в модель необходимо заполнить структуру данных типа TVariablesList, представляющую собой список изменяемых переменных и их значений. Пусть переменная такого типа имеет имя MyVariables, тогда сегмент кода программы, необходимый для определения значения этой переменной, имеет вид:

```
// задается число передаваемых переменных
MyVariables.Count:=1;
SetLength(MyVariables.Variables,1);
// задаются имена и значения передаваемых
переменных
MyVariables.Variables[0].Name:='Lambda';
MyVariables.Variables[0].Value:=inttostr(i);
```

Здесь под переменной *i* понимается значение интенсивности поступления заявок, которое изменяется внутри заданного пользователем интервала с шагом 1.

Для запуска одного прогона модели используется последовательность действий, которая записана в следующем фрагменте текста программы.

```
//Открыть GPSSW
RunGPSS(GPSSW_Path);
//Изменение модели
InsertModelin-
Data(ModelFileName,GPSSModelFileName,MyVa
riables);
//Выполнение модели
RuModel(GPSSModelFileName,ResultFileName);
//Извлечение данных из отчета
ExtractModeling-
Data(ResultFileName,MyReport);
//Закрытие GPSSW
StopGPSS;
//Удалить временные файлы
AssignFile(txt,GPSSModelFileName);
Erase(txt);
AssignFile(txt,ResultFileName);
Erase(txt);
```

Результат выполнения прогона модели (содержимое отчета) фиксируется в переменной MyReport типа TReport. Из этой переменной необходимо извлечь значение средней длины очереди СМО и перейти к реализации следующего прогона модели.

В результате прогонов модели был получен искомый график и данные (рис. 2), которые представлены в табл. 2.

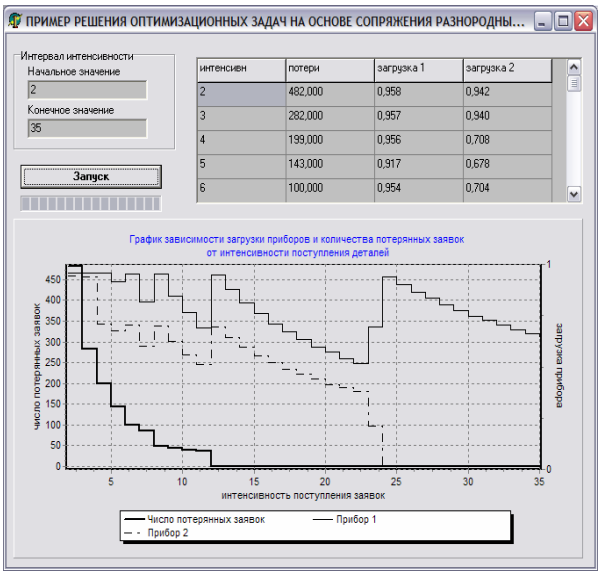

Рис. 2. Результаты решения задачи оптимизации работы конвейерной линии

Таблица 2

Решение задачи оптимизации работы конвейерной линии (α=0,5; β=0,001)

№ шага	$F(c,\rho_1\rho_2)$	λ	ρ_1	ρ_2	W_1	W_2
1	0,468	2	0,958	0,942	1,9	482
2	0,667	3	0,957	0,940	1,897	282
3	0,633	4	0,956	0,708	1,664	199
4	0,655	5	0,917	0,678	1,595	143
5	0,729	6	0,954	0,704	1,658	100
6	0,626	7	0,819	0,603	1,422	85
7	0,778	8	0,953	0,701	1,654	49
8	0,691	9	0,846	0,623	1,469	44
9	0,623	10	0,764	0,561	1,325	39
10	0,564	11	0,691	0,510	1,201	36
11	**0,821**	**12**	**0,949**	**0,694**	**1,643**	**0**
12	0,759	13	0,877	0,641	1,518	0
13	0,704	14	0,813	0,595	1,408	0
14	0,656	15	0,760	0,552	1,312	0
15	0,616	16	0,709	0,523	1,232	0
16	0,580	17	0,671	0,490	1,161	0
17	0,548	18	0,633	0,463	1,096	0
18	0,518	19	0,597	0,439	1,036	0
19	0,492	20	0,573	0,411	0,984	0
20	0,468	21	0,539	0,397	0,936	0
31	0,447	22	0,517	0,378	0,895	0
32	0,453	23	0,693	0,212	0,905	0
33	0,469	24	0,939	0	0,939	0
34	0,450	25	0,901	0	0,901	0
35	0,433	26	0,866	0	0,866	0
36	0,417	27	0,834	0	0,834	0
37	0,403	28	0,805	0	0,805	0
38	0,383	29	0,766	0	0,766	0
39	0,374	30	0,748	0	0,748	0
40	0,364	31	0,728	0	0,728	0
41	0,351	32	0,703	0	0,703	0
42	0,341	33	0,681	0	0,681	0
43	0,330	34	0,660	0	0,660	0
44	0,321	35	0,641	0	0,641	0

Из полученных данных видно, что оптимальной, для рассматриваемой задачи, будет интенсивность 12, так как при ней достигаются достаточно высокие коэффициенты загрузки устройств и в то же время заявки не теряются (значение целевой функции максимально и равно 0,821).

Проанализировав изложенную в этой статье информацию, можно сделать вывод о том, что предлагаемый нами подход позволяет предложить эффективное инструментальное средство решения оптимизационных задач.

Посредством сопряжения GPSSW и Delphi был получен программный комплекс, позволяющий использовать библиотеки моделей реализованных на языке GPSS, а также использовать множество готовых оптимизационных методов, реализованных на языке Pascal.

У сопряженной системы присутствуют следующие достоинства: лаконичность специализированного языка имитационного моделирования GPSSW, весь широкий спектр возможностей по созданию интерфейса, анимации процесса моделирования, импорта и экспорта данных, управления процессом моделирования и т.п. (от Delphi)

Таким образом. предлагаемый подход был успешно внедрен при построении сопряженной системы имитационного моделирования DG.

Заключение

Целью статьи было рассмотрение и решение ряда вопросов, связанных с обеспечением гарантоспособности программного обеспечения на основе автоматизации разнородных программных сред. Разработанный подход был продемонстрирован на примере взаимодействия высокоуровневой среды программирования Borland Delphi 7 со специализированной средой имитационного моделирования GPSS World.

При использовании имитационного моделирования в процессе поддержки принятия решений часто возникает необходимость в многократной модификации параметров модели и обработке большого объема получаемых данных.

Разработанная система помогает избавиться от большого числа ошибок, вызванных психофизиологическими качествами оператора-исследователя. Предложенный подход повышает гарантоспособность программного обеспечения путем минимизации дефектов взаимодействия, вызванных наличием человеческого фактора. Планируется внедрить рассмотренный метод в систему поддержки принятия решения по управлению ресурсами в распределенных объектах критического применения.

Литература

1. Харченко В.С. Гарантоспособность и гарантоспособные системы: элементы методологии // Радиоэлектронные и компьютерные системы. – № 5 (17). – С. 7-19.

2. Одарущенко О.Н. Терминологические аспекты теории надежности программных средств // Радиоэлектронные и компьютерные системы. – № 2 (6). – С. 88-94.

3. Шрайбер Т. Дж. Моделирование на GPSS. – М.: Машиностроение, 1980. – 592 с.

4. Архангельский А.Я. Программирование в Delphi 7. – М.: Бином-Пресс, 2003. – 1152 с.

Поступила в редакцию 19.02.2008

Рецензент: д-р техн. наук, проф. В.И. Хаханов, Харьковский национальный университет радиоэлектроники, Харьков.