В.И. Горбаченко, С.Л. Зефиров, Д.В. Мякишев

(Пензенский государственный технический университет)

МОДЕЛИРОВАНИЕ РАСПРЕДЕЛЕННЫХ СИСТЕМ ОБРАБОТКИ ДАННЫХ С ПОМОЩЬЮ ЯЗЫКА MODULA-2

For purpose of distributed data processing systems simulation offers the approach based on formalization of the simulated system taken as an integration of communicating concurrent processes. Simulation package is realized in Top Speed Modula-2 environment. It includes monitor, providing simulated time mechanism and processes synchronization, and a number of modules, implementing procedures of processes interaction, external interruptions simulating, processors and storage working simulation, message streams forming, statistics collection, service function, dialogue interface with the end user.

Ведущей тенденцией в развитии современных информационных технологий является использование систем распределенной обработки информации, объединенных в локальную и/или глобальную сеть. При создании таких систем возникает необходимость в оценке и выборе проектных решений, определяющих архитектуру сети, ее топологию и временные характеристики. Наиболее универсальными методами решения задач подобного рода являются различные виды моделирования. В частности, временное поведение систем удобно исследовать с помощью имитационного моделирования.

Доминирующей чертой распределенных систем является концептуальный и физический параллелизм процессов обработки информации. Поэтому при построении имитационных моделей распределенных систем выбран процессный подход, основанный на описании работы моделируемой системы в виде совокупности взаимодействующих процессов [1]. Процессный подход удобен для моделирования, так как реальная система и модель описываются с использованием одних и тех же формализмов, исключается промежуточное описание системы, например, с использованием аппарата систем массового обслуживания или агрегатов. Этим, в частности, процессный подход выгодно отличается от транзактного подхода, реализованного в системе GPSS. Описания процессов в реальной системе и в модели практически совпадают. Существенное отличие состоит в необходимости поддерживать в модели механизм модельного времени, обеспечивать порядок активизации процессов и синхронизацию событий в модели. Программа моделирования, основанная на процессном подходе, в определенной степени представляет собой прототип программного обеспечения реальной системы.

В качестве инструментального языка моделирования весьма перспективным представляется язык Modula-2 [2]. Основным преимуществом языка Modula-2 при реализации процессорного способа моделирования является аппарат сопрограмм, поддерживающий взаимодействие параллельных процессов. Язык Modula-2 обеспечивает хорошую поддержку для формирования развитых структур данных, что важно для построения систем моделирования. Модульная структура

программ на Modula-2 обеспечивает удобный способ расширения языка средствами моделирования.

На основе концепции параллельных процессов в среде TopSpeed Modula-2 [3] реализован пакет программ Sinproc (сокращение от "Simulation Information Process"). Процессы модели строятся на основе процедур, имитирующих процессы реальной системы. С каждым процессом связано локальное время, причем локальное время изменяется с переменным шагом до следующего события в процессе. Модельное время равно минимальному локальному времени всех процессов. Процесс является активным и его описатель (дескриптор) находится в списках дескрипторов текущих процессов (рис.1), если его локальное время совпадает с модельным. В активном процессе локальное время изменяется в зависимости от времени обработки данных на различных устройствах. Процесс может быть задержан. После этого описатель процесса помещается в список дескрипторов измененных процессов и производится активизация управляющего процесса (монитора). Монитор сортирует по времени и приоритету список готовых к выполнению процессов, выбирает и активизирует текущий активный процесс. Процесс, ожидающий выполне-

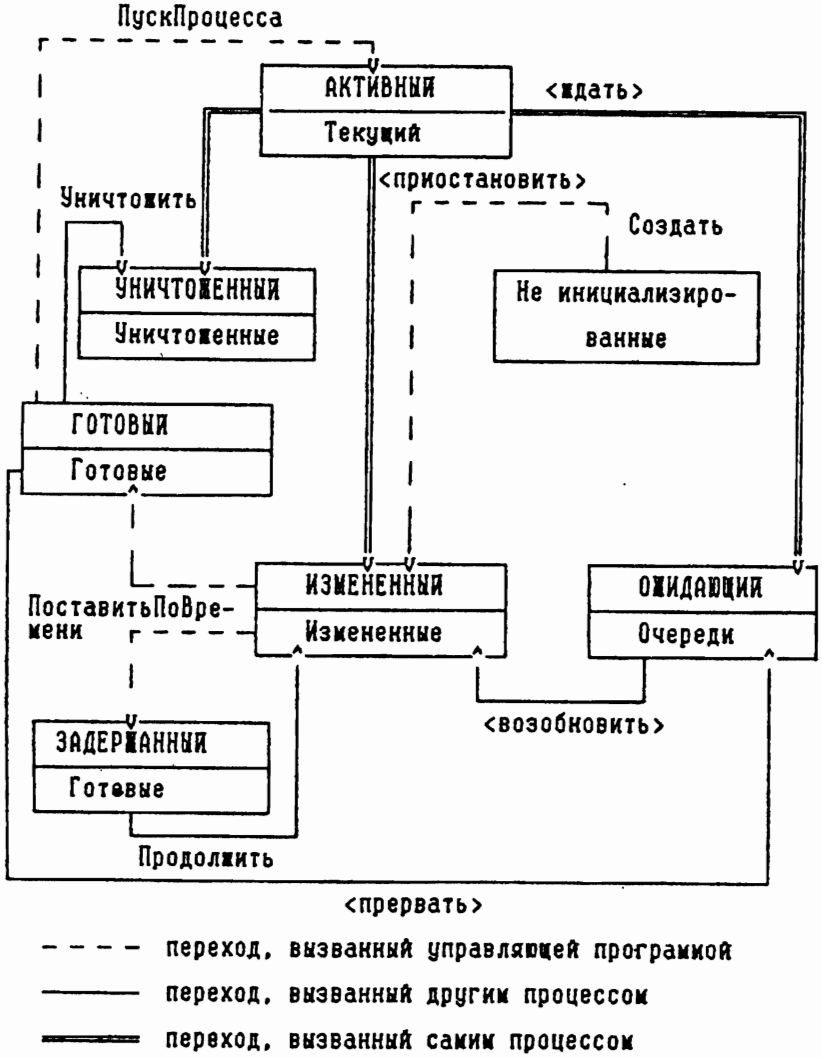

Рис. 1. Переходы между состояниями процессов.

ния какого-либо условия, является ожидающим и находится в одной из очередей процессов, ждущих освобождения ресурсов, или в одной из очередей прерванных процессов. Надписи у стрелок на рис.1 являются названиями процедур или совокупностей процедур (на рис.1 они заключены в скобки), которые вызывают соответствующий переход. Управление от монитора к процессу (Пуск Процесса) и обратно (Пуск Монитора) передается с помощью процедур Sinproc, основанных на процедуре TRANSFER языка Modula-2.

Синхронизация процессов в Sinproc осуществляется управляющей программой с использованием следующих средств: событий (процессы могут генерировать события и ждать события); задержек текущего процесса или других процессов на определенное и неопределенное время; принудительной установки времени активизации текущего или другого процесса; критических участков (если процесс вошел в критический участок, то другие процессы не могут войти в этот критический участок); прерываний (процесс может прервать работу другого процесса).

В Sinproc имитируется работа таких элементов информационных систем, как серверы (любые устройства обработки данных, в частности, процессоры), каналы ввода—вывода, память, буферы, очереди. Серверы являются разделяемыми ресурсами, доступ к которым поддерживается с помощью флагов, прерываний и очередей.

Входные сообщения (заявки) имитируются в Sinproc потоками сообщений: регулярными, простейшими и нормированными потоками Эрланга. Сообщения в потоке характеризуются не только временем поступления, но типом и длиной. Генераторы сообщений в моделирующей программе представляют собой процессы, считывающие сообщения из файлов заранее сформированных потоков сообщений. Формирование сообщений производится в Sinproc в диалоговом режиме. Особенностью Sinproc является возможность формировать входные потоки путем слияния нескольких потоков. Таким образом можно формировать потоки сложной структуры, описывающие поступление заявок от нескольких источников.

Средствами Sinproc производится сбор статистики по процессам (число задержек, общее время работы и задержек процесса); по серверам (число обращений, среднее время и дисперсия времени работы, загрузка сервера); по памяти (число обращений, загрузка, среднее значение и дисперсия времени работы, максимальный занятый объем и максимальное время пребывания сообщений в памяти); по очередям (число обращений, количество потерянных сообщений, максимальное время пребывания сообщения в очереди и максимальная длина очереди); по сообщениям каждого типа (среднее время существования сообщений, число обработанных и потерянных сообщений); по генераторам сообщений (количество сгенерированных сообщений). Имеется также статистика пользователя, позволяющая получить математическое ожидание и дисперсию произвольных переменных модели.

ППП Sinproc реализован в виде набора модулей языка Modula-2 (модули русифицированы). В состав Sinproc входят модули управления моделированием, низкоуровневых операций над процессами, синхронизации процессов, имитации элементов систем, пользовательской статистики, организации системы моделирования и вспомогательные.

Моделирующая программа пользователя (рис.2) строится по правилам языка Modula-2. В разделе "Пользовательские процедуры процессов" с использованием процедур Sinproc и операторов языка Modula-2 описываются процедуры модели. С помощью процедуры "Стандартная

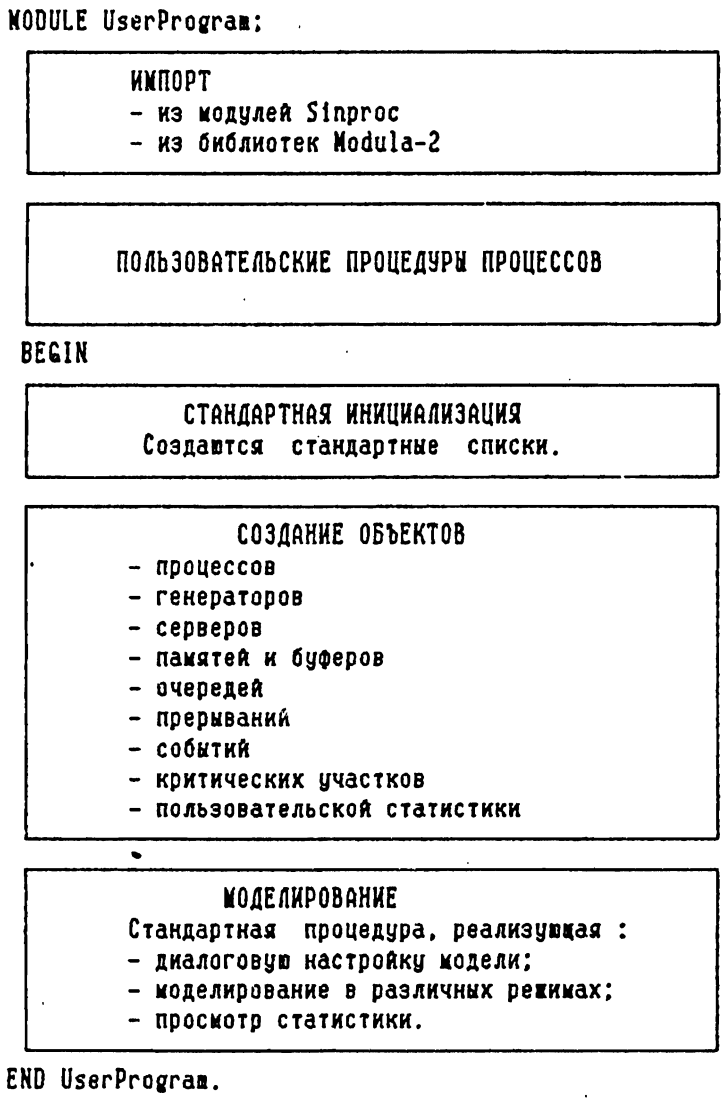

Рис. 2. Структура пользовательской программы.

Инициализация" создаются списки объектов модели. В разделе "Создания объектов" пользователь, используя процедуры создания и инициализации объектов, создает из процедур, описанных во втором разделе, процессы и инициализирует все элементы модели. Далее может следовать раздел настройки модели, в котором устанавливаются параметры модели (параметры можно установить и изменить в режиме диалога). Процедура "Моделирование" реализует процесс моделирования и просмотр статистики.

Среда ППП Sinproc позволяет в диалоговом режиме формировать и просматривать потоки входных сообщений, задавать исходные данные и режимы моделирования, просматривать статистику. Кроме автоматического выполнения моделирования, предусмотрены следующие режимы: трассировка (выводится имя текущего процесса); пошаговое выполнение (отображаются состояния процессов, переход к очередному событию происходит по клавише <Enter>); контрольная точка по заданному значению модельного времени; принудительное завершение моделирования по клавише <Esc>.

Для демонстрации возможностей ППП Sinproc рассмотрим моделирование следующей системы (рис.3). Процессор ЦП1 получает сообщения

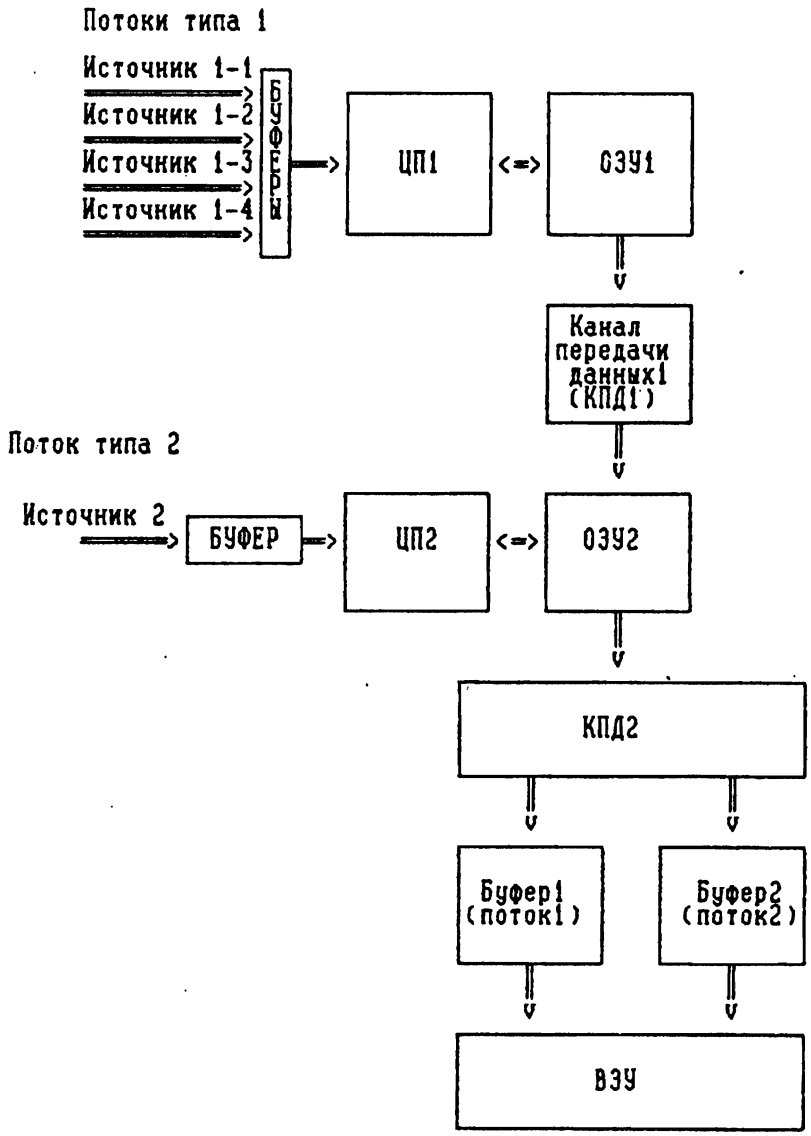

Рис. 3. Структура моделируемой системы.

одного типа от четырех источников, блокирует поступившие сообщения и передает блоки сообщений на обработку процессору ЦП2, который принимает блоки сообщений, обрабатывает каждое сообщение блока, буферизует обработанные сообщения, выводит сообщения из буфера во внешнее запоминающее устройство (ВЗУ). Дополнительно ЦП2 принимает сообщения потока второго типа, обрабатывает их, буферизует и выводит в то же ВЗУ. Сообщения, переписанные в ВЗУ, считаются обработанными и удаляются из модели.

Модель в виде схемы взаимодействия процессов представлена на рис.4. На схеме в виде прямоугольников с соответствующими именами показаны процессы. Внутри процессов обозначены используемые ресурсы (серверы, каналы, память), обозначены также буферы для приема входных сообщений и генераторы сообщений. Разделяемый ресурс (ЦП2) показан во всех процессах, использующих этот ресурс. Передача информации показана двойными стрелками, посылка событий —одинарными с указанием названия события. Линии событий направлены от

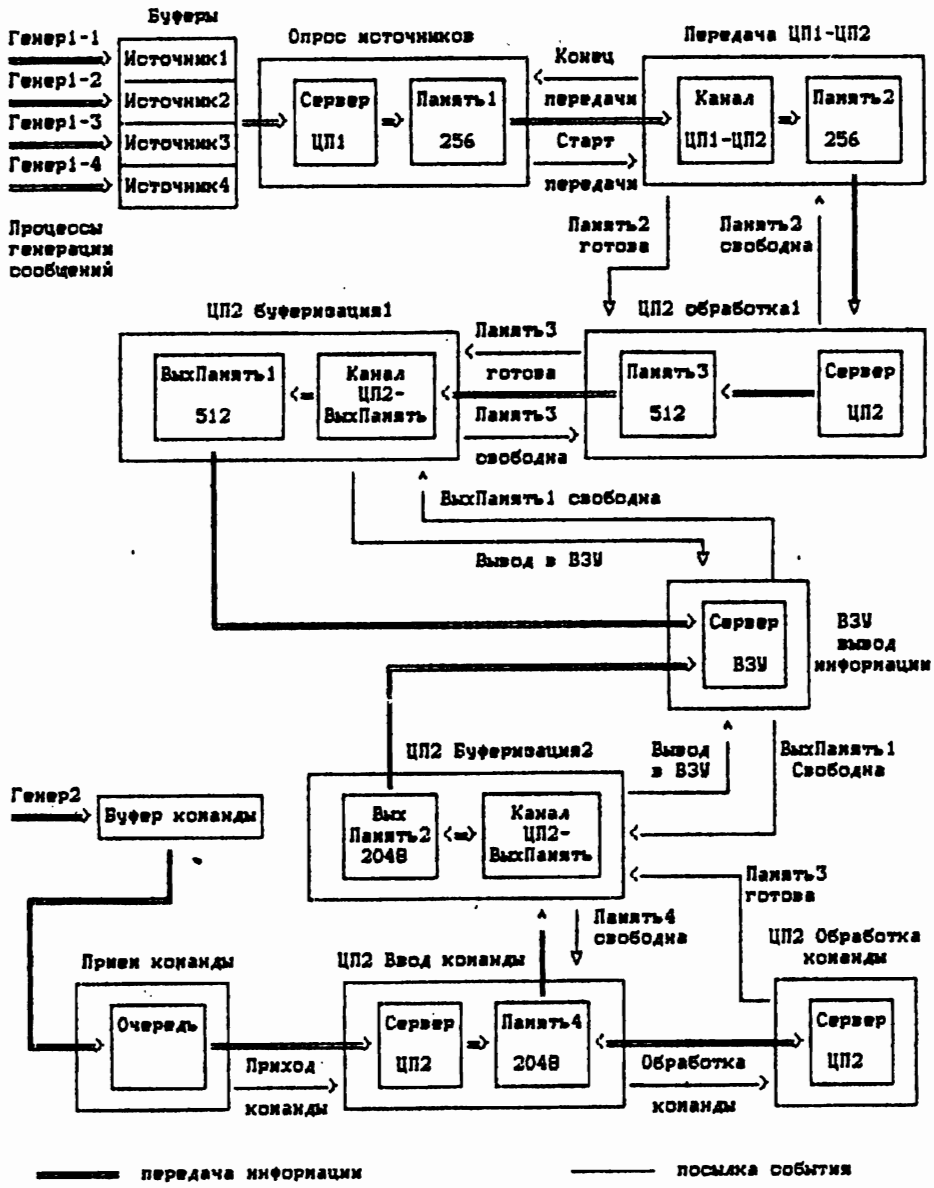

Рис. 4. Схема взаимодействия процессов примера.

процессов, генерирующих события, к процессам, управляемым этими событиями.

Структура моделирующей программы является стандартной для Sinproc. Программа содержит 5 процессов-генераторов, 8 пользовательских процессов, 13 событий, 1 критический участок, 3 сервера, 2 канала, 6 памятей, 6 буферов, 1 очередь. Все процессы строятся в виде бесконечных циклов (оператор LOOP) и синхронизируются посредством событий. В модели используется критический участок, чтобы процессы "Опрос источников" и "Передача ЦП1-ЦП2", выполняемые на ЦП2, не происходили одновременно. Из одной и той же процедуры может быть создано несколько процессов, например, из процедуры g1 создано четыре процесса приема сообщений от разных генераторов. Фрагмент моделирующей программы представлен на рис. 5.

```
                Процесс : 'ЦП2_Обработка1' .
     Процесс обработки блока сообщений на процессоре 'ЦП2'.

     Процесс активизируется по событию 'Пам2Готова'
   / ready_мем2 / . Если память 'Память3' непуста, то процесс
   будет ждать событие 'Пам3Свободна' / free_мем3 / .
     Если же память 'Память3' свободна или событие free_мем3
   получено, то сообщения по одному считываются из памяти
   'Память2' в буфер 'БуфЦП2' и обрабатываются на 'ЦП2'.
   Длительность обработки одного сообщения 1мС. При обработке
   длина сообщения увеличивается на 6 байт. После обработки
   сообщение из 'БуфЦП2' переписывается в память 'Память3'.
     По завершении обработки блока посылается события :
   'Пам3Готова' / ready_мем3 /, по которому активизируется
   процесс 'ЦП2_Буферизация1' и 'Пам2Свободна' / free_мем2 /.
     Затем процесс 'ЦП2_Обработка1' задерживается до
   получения очередного события 'Пам2Готова' / ready_мем2 /.
*)

PROCEDURE pr21;
VAR v:CARDINAL;
    b,b1,b2,b3:BOOLEAN;
BEGIN
  LOOP
    ЖдатьОчередноеСобытие(ready_мем2,FALSE);
    СостояниеПамяти('Память3',b1,b2,b3,v);
     IF b1 THEN ЖдатьОчередноеСобытие(free_мем3,FALSE); END;
    СостояниеПамяти('Память2',b1,b2,b3,v);
      WHILE b1 DO
        b:=ЗахватитьСервер('ЦП2');
        b:=ЧтениеИзПамяти('Память2','БуфЦП2');
        РаботаНаСервере('ЦП2',1.);
        ИзмДлинуСообщ('БуфЦП2',6);
        b:=ЗаписьВПамять('БуфЦП2','Память3');
        ОсвободитьСервер('ЦП2');
        СостояниеПамяти('Память2',b1,b2,b3,v);
      END;
      ПослатьСобытие(free_мем2);
      ПослатьСобытие(ready_мем3);
  END;
END pr21;
```

Рис. 5. Фрагмент моделирующей программы.

С помощью разработанного пакета можно проводить дискретно-событийное моделирование локальных сетей, многопроцессорных систем реального времени и однопроцессорных вычислительных комплексов, поддерживающих мультипрограммный режим. Для этого необходимы система программирования Top Speed Modula-2 и пакет Sinproc.

ЛИТЕРАТУРА

1. *Хоар Ч.* Взаимодействующие последовательные процессы. М., 1989.

2. *Prashant W., Sztipanovits J.* // Journ. of Pascal, Ada & Modula-2. 1990. Vol.9., N4. P.20, 22—29, 32—33.

3. *King K.N.* TopSpeed Modula-2. Language Tutorial. London, 1991.

Рукопись поступила 25.10.93