

## ВИЗУАЛИЗАЦИЯ МОДЕЛЕЙ ПРОЦЕССОВ, УПРАВЛЯЕМЫХ СООБЩЕНИЯМИ, В СИСТЕМЕ GRAPHPLUS TEMPLET

**С.В. Востокин**

Самарский государственный аэрокосмический университет им. акад. С.П. Королева  
443086, г. Самара, Московское шоссе, 34

*Рассматривается способ визуализации моделей процессов, управляемых сообщениями. Приведен пример параллельной программы типа «разветвление – слияние».*

**Ключевые слова:** визуальное программирование, модель процессов, параллельное программирование.

При разработке параллельных и распределенных программ часто удобнее структурировать код в терминах моделей процессов, управляемых сообщениями. В моделях этого типа естественно представляется параллелизм, что позволяет применять их для программирования многопроцессорных компьютеров. Они подходят для программирования как в общей, так и в распределенной памяти; проще в использовании, чем аппаратно ориентированные модели многопоточных вычислений и обмена сообщениями в API операционных систем. В то же время применение объектно-ориентированных языков, например C++, C#, Java, для выражения семантики моделей процессов, управляемых сообщениями, приводит к ряду трудностей, проистекающих из процедурной семантики вызовов методов в перечисленных языках.

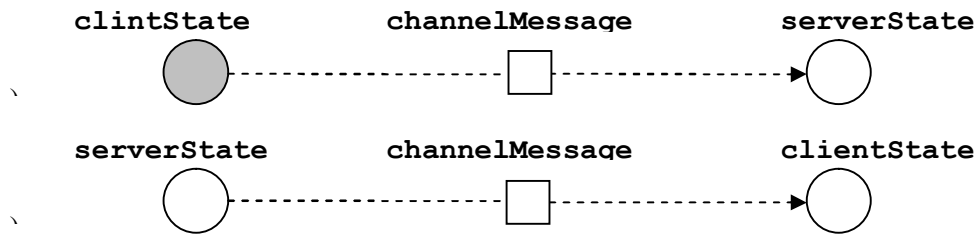
Для преодоления указанного недостатка традиционных объектно-ориентированных языков в разрабатываемой нами системе Graphplus templet используется визуальное проектирование процессов и каналов – интерфейсов обмена сообщениями между процессами.

Базовыми концепциями визуального языка являются типы сообщений (типы данных, определяемых пользователем) и методы (процедуры), выполняющие действия над переменными указанных типов, передаваемых как параметры вызова. Типы и методы кодируются обычным образом на языках C++, C#, Java и не содержат обращений к примитивам синхронизации или обмена сообщениями.

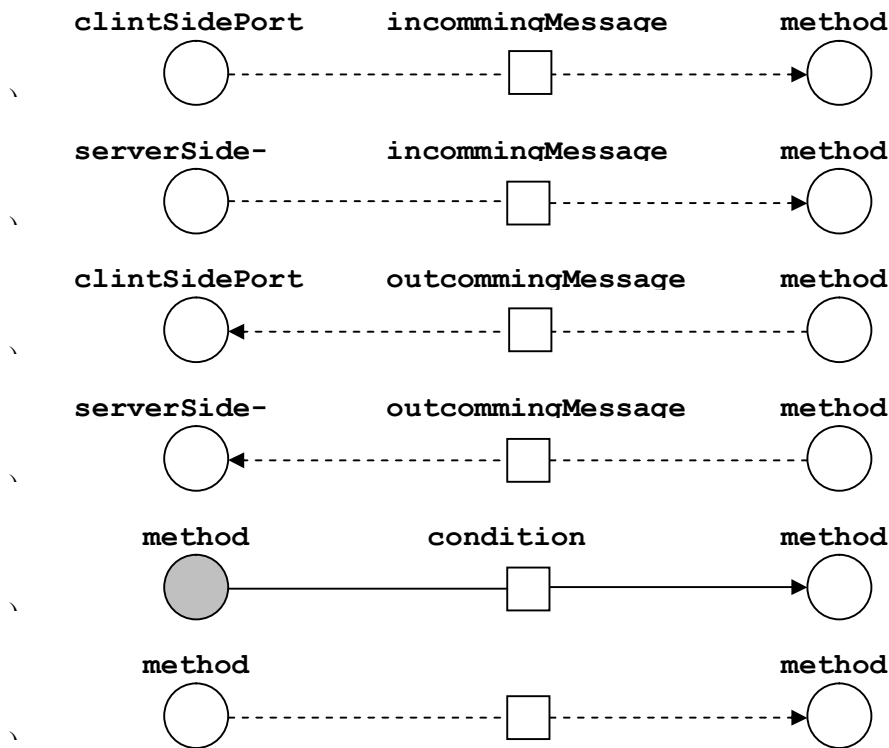
Объект-канал описывает протокол обмена сообщениями между парой процессов, подключенных к нему. Канал представляется ориентированным аннотированным графом. Вершины графа имеют пометки типа «клиент» и «сервер», обозначающие направление передачи сообщения. Пометки дуг адресуют тип передаваемого сообщения. Одна из вершин типа «клиент» имеет пометку «исходная», обозначающую начальное состояние. Канал работает как конечный автомат. Используемые для представления каналов графические примитивы показаны на рис. 1.

Объект-процесс описывает алгоритм обработки поступающих сообщений. Процесс также представляет собой ориентированный аннотированный граф. Вершины графа относятся к двум типам: порты и методы. Порты – это точки подключения каналов, они дополнительно снабжены пометками «клиент» или «сервер». Порты идентифицируются именами и типами каналов. Вершины-методы идентифицируются именами. Вершины графа-процесса связаны дугами двух типов: описывающих передачу управления и передачу сообщений из каналов в методы. На рис. 2 дуги *а*, *б*,

*δ*, *ε* описывают возможные связи между вершинами по данным, а дуги *δ* и *ε* – связи по управлению.

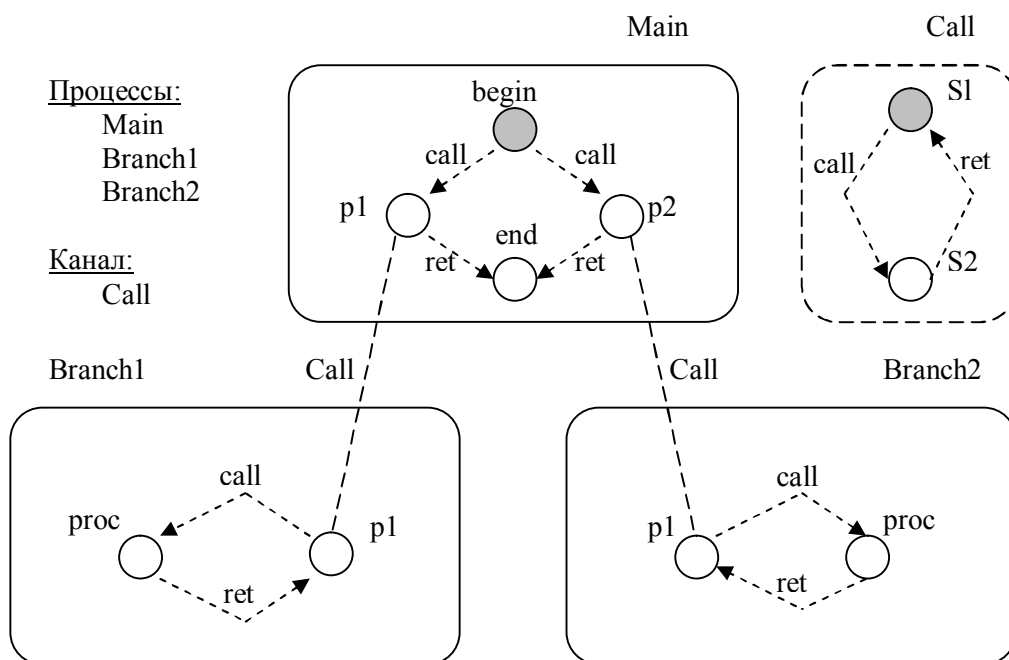


Р и с. 1. Графические символы, используемые для описания каналов



Р и с. 2. Графические символы, используемые для описания процессов.

Пример использования описанных выше графических обозначений приведен на рис. 3.



Р и с. 3. Модель параллельной программы

На рис. 3 представлена программа управления параллельными вычислениями типа «разветвление – слияние». Здесь Main – это главный процесс, запускающий вычисления, о чем говорит пометка его вершины begin. Метод begin имеет два выходных параметра типа call, соответствующих исходящим из его вершины дугам. Сообщения call передаются по каналам типа Call, связывающих процессы, и поступают на серверные порты p1, а затем в методы proc процессов Branch1 и Branch2. Выходными параметрами методов являются ответные сообщения ret, в конечном итоге поступающие в метод end процесса Main. Пометка 2 метода end говорит о том, что он запустится, только когда его достигнут 2 сообщения. Так как из метода end больше сообщений не отправляется, вычисления останавливаются.

Показанный пример, а также другие примеры параллельных управляющих алгоритмов размещены на сайте проекта «Графплюс» по адресу <http://graphplus.ssau.ru>. Модели параллельных программ [1], подобные модели, показанной на рис. 3, хранятся в XML-файлах. По файлам специальным генератором строится код на языке C++. Данный код повторяет структуру графов рис. 3: вершины графа соответствуют case-секции инструкции switch в обработчиках сообщений процессов. Поэтому сгенерированный код понятен и удобен для отладки. Также видны преимущества графического кодирования – помимо наглядности, графическая нотация оказывается лаконичнее сгенерированного кода на языке C++. Описанный метод визуализации основан на парадигме автоматного программирования [2]. Шаги автомата – это вызовы методов процесса. Достоинством данной графической нотации является явное определение контекста вызова метода и интерфейсов взаимодействия процессов. В отличие от других автоматных подходов, работа процессов в общем случае асинхронна.

Представленный визуальный язык применяется в системе автоматизации параллельного программирования Graphplus templet для суперкомпьютера «Сергей Коро-

лев», который установлен в Самарском государственном аэрокосмическом университете (национальном исследовательском университете). Дальнейшее развитие языка предполагает большую адаптацию к технологиям XML, таким как схемы XSL и трансформации XSLT, с целью использования стандартных средств визуализации современных XML-редакторов, в том числе поставляемых в комплектах интегрированных средств разработки.

#### БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Востокин С.В.* Графическая объектная модель параллельных процессов и ее применение в задачах численного моделирования. – Самара: Изд-во Самарского научного центра РАН, 2007. – 286 с.
2. *Поликарпова Н.И., Шалыто А.А.* Автоматное программирование. – СПб.: Питер, 2009. – 176 с.

*Статья поступила в редакцию 5 октября 2011 г.*

## VISUALIZATION OF MESSAGE-DRIVEN PROCESS MODELS IN GRAPHPLUS TEMPLAT SYSTEM

***S.V. Vostokin***

Samara State Aerospace University  
34, Moskovskoe sh., Samara, 443086

*The approach of visualization of message-driven process models is discussed. Sample fork-join parallel program presented.*

***Keywords:*** *visual programming, process model, parallel programming.*

---

*Sergey V. Vostokin (Dr. Sci. (Techn.)), Professor.*