УДК 004.422: 004.5

РЕСУРСНО-СЕТЕВАЯ МОДЕЛЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

К. т. н. А. С. Пригожев

Одесский национальный политехнический университет
Украина, г. Одесса
oleksandr@prigozhev.name

Проанализированы существующие архитектуры программного обеспечения и ресурсно-сетевые модели. Установлены типы потоковых моделей для моделирования программного кода и пользовательского интерфейса. Предложено усовершенствование известной графовой модели программы на основе известной ресурсно-сетевой модели Форда—Фалкерсона. Для моделирования процессов функционирования программного обеспечения в целом предложена новая потоковая модель — гибридная ресурсная сеть.

Ключевые слова: *тестирование ПО, модель Форда-Фалкерсона, однородная ресурсная сеть.*

Разработка большинства программных систем в настоящее время ориентирована на применение гибкой методологии разработки (Agile-методологии), основополагающие концепции которой выражены в манифесте Agile[1]:

— люди и их взаимодействие важнее процессов и инструментов;
— готовый продукт важнее документации по нему;
— сотрудничество с заказчиком важнее жестких контрактных ограничений;
— реакция на изменение важнее следования плану.

Это приводит к необходимости ускорения процессов тестирования при сохранении объемов выполнения тестов. Одним из путей решения данной проблемы является применение средств автоматизации тестирования. Современные приложения создаются на основе компонентной архитектуры, где каждый компонент системы может быть написан на своем языке и с использованием своих парадигм и подходов к программированию. Это делает затруднительным проведение автоматизированного тестирования программных систем.

В этих условиях актуальной является разработка специализированных сред тестирования, основанных на формальных моделях исходного кода и спецификаций программных систем. Перспективным в данной области тестирования представляется применение графовых моделей, в частности аппарата ресурсных сетей. Целью представляемого исследования является построение модели программного кода и пользовательского интерфейса, основанной на потоковой графовой модели. Для достижения данной цели решены следующие задачи:

— анализ существующих архитектур программного обеспечения и ресурсно-сетевых моделей;
— построение потоковой модели программного кода;
— построение ресурсно-сетевой модели интерфейса пользователя;
— построение многослойной ресурсно-сетевой модели программного обеспечения.

Объектом исследования являются современные потоковые модели программного обеспечения. Предметом исследования является применение потоковой модели для построения среды автоматизированного тестирования.

В современных потоковых моделях выделяют два основных подхода: модели Форда—Фалкерсона [2] и однородные ресурсные сети [3]. В обоих случаях модель представляет собой граф, в котором циркулирует некоторое количество дискретного ресурса. Каждому ребру графа потоковой модели приписана неотрицательная пропускная способность. Различие представленных моделей заключается в отсутствии в однородной ресурсно-сетевой модели вершин стока и истока ресурса. В однородной ресурсной сети загрузка ресурса происходит в любую вершину сети.

Перечисленные возможности делают возможными использование указанных типов сетей для моделирования программного обеспечения с различными типами интерфейсов. Для моделирования

программного кода удобно использовать модель Форда—Фалкерсона, а для моделирования пользовательских интерфейсов — однородную ресурсную сеть.

Модель Форда—Фалкерсона строится на основе основных алгоритмических конструкций: последовательного выполнения операторов, альтернативы и цикле. В [4] указанные операции представимы в виде многосортной алгебры [5] вида

$$((P,O),S)$$

где P — множество предикатов некоторой предметной области, O — множество операций, S — сигнатура, состоящая из композиции, альтернативы и цикла.

Основные алгоритмические структуры определены в (1) как многосортные операции [5] различных типов. Композиция двух операций — это многосортная операция, определяемая следующим соответствием: $\varphi : O^2 \to O$. Тип данной многосортной операции (2,2,2). Многосортная операция альтернатива задается соответствием вида $\psi : P \times O \to O$, а ее тип (2,1,2). Операция цикла представляется соответствием вида $\alpha : P \times O \to O$. Тип этой операции (2,1,2). Согласно [5], цифры в кортежах, обозначающих типы операций, обозначают номера основ многосортной алгебры (1), причем первая компонента кортежа указывает номер основы, которой принадлежит результат многосортной операции. Далее перечисляются номера основ, которым принадлежат аргументы многосортной операции, причем порядок следования номеров соответствует порядку следования множеств-основ в операции декартова произведения, соответствующей области отправления соответствия.

Учитывая вышеприведенные определения, а также определения операций и предикатов как соответствий, можно сделать вывод о перспективности применения графовых потоковых моделей для писания программного кода. Ресурсом в данной сети будет являться тест-пакет – набор переменных анализируемой программы. Фрагмент тест-пакета для одной переменной приведен на рис. 1.

Имя переменной	Видимость переменной для слоя	Тип переменной	Минимально возможное значение исходя из типа переменной	Максимально возможное значение исходя из типа переменной	Минимально возможое значение по задаче	Максимально возможное значение по задаче

Рис. 1. Формат тест-пакета

Все операции, включая и алгоритмические конструкции, отображаются ребрами графа, а вершины графа, ограничивающие ребра играют роль памяти модели. Пропускная способность каждого ребра графа выбирается из множества $\{0;1\}$. В случае операции композиции φ, а также некоторой операции из множества O пропускная способность ребра, моделирующего эти операции равна 1. В случае использования оператора альтернативы, пропускные способности ребер показаны на рис. 2.

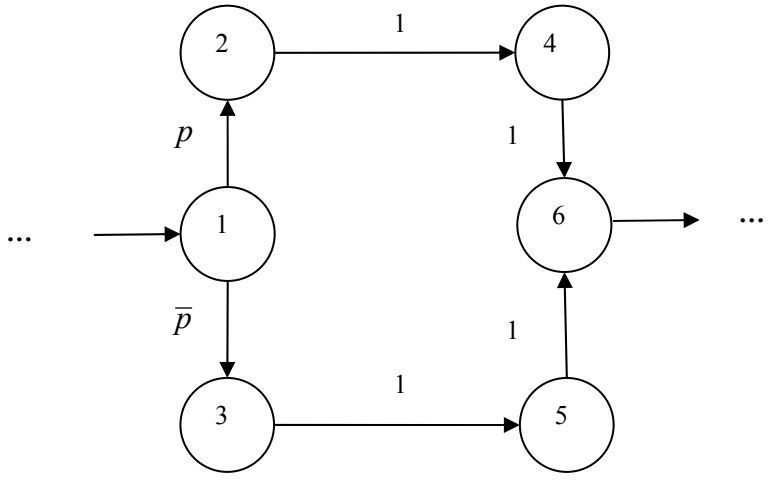

Рис. 2. Фрагмент модели Форда—Фалкерсона для операции альтернатива

На рис. 2 вершина *1* является входной вершиной для оператора альтернативы. Ребро *1—2* моделирует переход по условию «истина», ребро *1*—3 моделирует переход по условию «ложь». Ребро *2—4* моделирует выполнение оператора, выполняющегося при истинном условии альтернативы, ребро *3—5* моделирует выполнение оператора, находящегося в ложной ветке альтернативы. Вершина *6* является выходной вершиной для операции альтернативы.

Величина пропускной способности ребер *1—2* и *1*—3 вычисляется согласно логическому выражению, сопоставленному данным ребрам. При этом предикат $p \in P$ принадлежит второй основе алгебры (см. формулу (1)).

Простейшая операция цикла α согласно своей записи представима в виде графа, как показано на рис. 3/

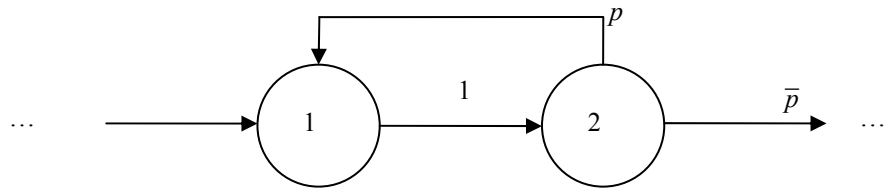

Рис. 3. Фрагмент графовой модели простого цикла с постусловием

На рис. 3 переход ребро от вершины *1* к вершине *2*, которому приписана 1 пропускная способность, означает выполнение оператора цикла. Ребро от вершины *2* в вершину *1*, которому сопоставлен предикат *p*, осуществляет проверку условия и возврат на начало цикла. Выход из цикла происходит по ребру, промаркированному логическим выражением $\overline{p}$.

Любая операция или функция, сколь угодно сложная, может быть представлена в виде отношения или соответствия, т. е. подмножества декартова произведения. Из этого следует, что выполнение любой операции, отличной от характеризуемых многосортной алгеброй (1), можно свести к задаче поиска значений в таблице операций и изменения значений в тест-пакете при прохождении через ребро.

Таким образом, построенный граф можно использовать в качестве ресурсной модели Форда—Фалкерсона для программного кода. Данная модель является дальнейшим развитием известной модели графа исполнения программы, приведенной в [6], в части формализации процесса анализа графа и изменения тестовых значений.

Модель пользовательского интерфейса строится на основе однородной ресурсной сети. Ребра и вершины графа в этом случае играют ту же роль, что и в рассмотренной ранее модели Форда—Фалкерсона. Пропускные способности ребер выбираются из множества {0;1} в зависимости от вероятности перехода пользователя по ребру. Такая модель позволяет расширить подход на основе графов, применяемый для моделирования программных системы на моделирование интерфейсов пользователя.

На основе анализа современных архитектур программного обеспечения установлено, что любое программное обеспечение можно представить в виде четырех уровней: структурный уровень, уровень элементов, уровень событий, уровень реализации. На структурном уровне определяются взаимосвязи структурных единиц пользовательского интерфейса. С точки зрения командного интерфейса в роли структурных единиц интерфейса выступает команда пользовательского интерфейса. В графическом интерфейсе структурной единицей является элемент управления контейнер, содержащий другие элементы управления. В случае использования специализированных интерфейсов, например, звуковых [7], структурной единицей интерфейса будет являться элемент интерфейса, в который вложены остальные элементы. В случае, приведенном в [7], это «аудиокомната».

Уровень элементов представляет собой описание взаимосвязей элементов внутри конкретного контейнера. Примерами элементов внутри контейнера являются: для командного интерфейса — параметр команды, для графического интерфейса — элемент управления, отличный от элемента-контейнера, для специализированных интерфейсов — некоторая минимальная единица интерфейса, которая содержит только данные или команды пользователя.

Уровень событий описывает взаимосвязь событий элементов интерфейса между собой. Уровень событий присутствует исключительно в событийно-ориентированных интерфейсах и представляет собой описание последовательностей событий, происходящих в интерфейсе пользователя.

Уровень реализации — это формальное описание программного кода приложения в виде некоторой формальной спецификации.

С учетом перечисленных особенностей архитектуры, для моделирования программного обеспечения предлагается использовать новую потоковую модель — многослойную ресурсную сеть. Базовым компонентом данной модели является слой, которому сопоставлена либо однородная ресурсная сеть с одной вершиной-стоком, либо потоковая модель Форда—Фалкерсона. Слои связаны между собой коммуникационными слоями. Коммуникационный слой содержит простейшую модель Форда—Фалкерсона. Вершиной-истоком этой модели является вершина-сток некоторого слоя, а вершиной-стоком будет являться вершина-исток некоторого другого слоя.

Каждый из перечисленных уровней архитектуры программного обеспечения может быть смоделирован одним или несколькими слоями, которые связаны между собой коммуникационными слоями. Если слой соответствует структурному уровню, уровню элементов или уровеню событий, то ему, как правило, соответствует однородная ресурсная сеть, а слою, который описывает уровень реализации — модель Форда—Фалкерсона. Ресурсом в гибридной ресурсной сети является тест-пакет (см. рис. 1).

Таким образом, получена новая потоковая модель — гибридная ресурсная сеть, которая является интеграцией известных моделей Форда—Фалкерсона и однородной ресурсной сети. Получившая дальнейшее развитие в части использования для моделирования ресурсной сети Форда—Фалкерсона графовая модель программного кода позволяет формализовать процесс анализа функционирования программы.

ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ

1. Charles Cobb Making Sense of Agile Project Management: Balancing Control and Agility .— New Jersey: Wiley, 2011 .
2. Форд Л.Р., Фалкерсон Д. Потоки в сетях .— Москва:Мир, 1966
3. Кузнецов О. П. Однородные ресурсные сети I. Полные графы // Автоматика и телемеханика.— 2009.— № 11 .— С. 136–147
4. Цейтлин Г. Е. Введение в алгоритмику .— Киев: Сфера, 1998
5. Белоусов А. И. Ткачев С. Т. Дискретная математика.— Москва: Изд. МГТУ им. Н. Э. Баумана, 2004.
6. Бейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. — С.-Петербург: Питер, 2004.
7. Mynatt E. Transforming graphical interfaces into auditory interfaces for blind users // Human-Computer Interaction .— 1997 .— N 12 .— P. 7—45

———————

O. S. Prygozhev

Resource-network model of software.

The author analyzes the existing software architectures and resource-network models and determines the types of flow-oriented models for the code and user interface simulation. The paper presents the enhancement of the well-known program graph model based on the known Ford—Fulkerson resource-network model. For modeling of processes of functioning of the software as a whole, a new flow-oriented model is offered — a hybrid resource network.

Keywords: *software testing, Ford-Fulkerson model, uniform resource network.*