В.В. Пекунов

Ивановский государственный энергетический университет

ТЕОРИЯ ОБЪЕКТНО-СОБЫТИЙНОГО МОДЕЛИРОВАНИЯ ПОСЛЕДОВАТЕЛЬНЫХ И ПАРАЛЛЕЛЬНЫХ ПРОЦЕССОВ. ОСНОВНЫЕ СЛЕДСТВИЯ И ПРИЛОЖЕНИЯ В ПРОГРАММИРОВАНИИ, МОДЕЛИРОВАНИИ И ПОРОЖДЕНИИ ПРОГРАММ

В данной работе на основе самых общих предположений выводится теория объектно-событийных моделей (ОСМ) последовательных и параллельных процессов и определяются ее наиболее перспективные приложения.

Лемма А1. В процессе исполнения любой последовательной или параллельной программы любая вычислительная система (машина) осуществляет серию переходов между звеньями обработки, отображаемую трассой исполнения. Одноименным, однозначно идентифицируемым по имени элементам трассы исполнения соответствует факт прохождения системы через одно и то же звено обработки. Каждому этапу трассы соответствует один или несколько (обрабатываемых параллельно на одном и том же интервале времени) элементов. Любое звено реализует параметризованный алгоритм, обрабатывающий внутренние переменные своего состояния и/или внешние данные. Его исполнение определяется комплексом: а) глобальных данных вычислительной системы; б) параметров вызова, которые могут быть реализованы в рамках глобальных данных; в) значений внутренних переменных. Иных побочных эффектов нет.

Доказательство. Любая программа, реализующая вычислимый по Тьюрингу алгоритм, может быть реализована с помощью машины Тьюринга (МТ) или комплекса таких машин. Для МТ всегда можно построить трассу исполнения, которая может быть однозначно отображена в трассу любой иной последовательной вычислительной системы. Элементами трассы МТ будут имена состояний, однозначно соответствующие уникальным звеньям обработки. Для комплекса МТ также можно построить

общую трассу, комбинируя их частные трассы по принципу одновременности исполнения.

Алгоритм звена обработки в предельном случае соответствует фрагменту программы переходов из одного состояния МТ. Его работа зависит от глобального контекста (ленты) и внутреннего состояния (положения головки).

Утверждение У1. Модель процесса исполнения, адекватно отражающая логику переходов исходной системы между элементами трассы исполнения, каждый из которых подразумевает параметризованный вызов соответствующего алгоритма, осуществляющего преобразования внутренних и внешних данных, тождественные преобразованиям, выполняемым исходной системой при прохождении через указанные элементы трассы исполнения, является моделью исходной программы.

Доказательство. Вытекает из А1.

Утверждение У2. Назовем факт прохождения программы через один или несколько этапов трассы ее исполнения *событием*. Процесс обработки события означает исполнение этих и только этих этапов. В контексте календаря событиям соответствуют непересекающиеся интервалы времени обработки. Между этими событиями существует строгий порядок следования, соответствующий порядку прохождения трассы.

Доказательство. Непересекаемость интервалов следует из А1, поскольку этапы трассы исполняются последовательно. Строгий порядок событий следует из отношения строгого порядка интервалов времени.

Следствие У2.1. Априорно считаем параллельным исполнение этапов трассы, соответствующих одному событию. Для установления последовательности их исполнения необходимы средства, устанавливающие очередность прохождения.

Лемма Л1. Параметризованный алгоритм А, вызываемый при прохождении программы через одноименные элементы трассы исполнения, обладающий внутренним состоянием, реализуется объектом некоего класса С. Каждому факту прохождения соответствует событие, приводящее к вызову отдельного метода класса С.

Доказательство.

1. Возможность сопоставления события отдельному методу следует из А1, У2 и определения метода в рамках объектно ориентированного подхода. Метод интерпретируется как фрагмент алгоритма звена обработки, активный в контексте события.

2. Если одному или нескольким различным событиям соответствуют различные алгоритмы обработки, то их реализация в отдельных методах строго логична. В случае единого алгоритма достаточно прибегнуть к копированию методов.

3. Внутреннее состояние алгоритма А и производных от него алгоритмов представляется полями класса С.

4. Параметризация вызовов исходного алгоритма А, таким образом сводится к параметризации эквивалентных им по действию вызовов методов класса С.

Базовая теорема ОСМ (ТМ1).

А. Модель исходной последовательной или параллельной программы, может быть представлена графово-событийной схемой. Узлами графа являются объекты, реагирующие на события параметризованными вызовами методов, модифицирующих состояние объекта и глобальные данные. События происходят в порядке, определяемом их календарем.

Б. Если в ответ на одно событие вызываются методы нескольких или даже всех объектов, то необходимо специфицировать порядок их последовательного или параллельного исполнения. Если для двух объектов определено отношение строгого порядка следования, их методы исполняются последовательно. Иначе параллельно. Отношение задано дугами графа.

В. Граф должен быть связным и ациклическим.

Доказательство.

А. Следует из У1, У2, Л1

Б. Следует из самого утверждения.

В. Любая программа имеет единственные начальную и конечную точки. Реагировать на одно и то же событие потенциально могут все узлы. Любой промежуточный узел находится

в отношении строгого следования по отношению, по меньшей мере, к этим двум точкам. Это антирефлексивное, транзитивное и антисимметричное отношение. Следовательно, граф является связным и сетевым, то есть ациклическим.

Лемма Л2. Узел, имеющий несколько исходящих дуг в узлы, инцидентные по входам только ему, в зависимости от событийной схемы является точкой принятия решения о переходе последовательной программы и/или точкой порождения параллельных ветвей.

Доказательство. Если следующие узлы некоторых исходящих ветвей предусматривают реакцию на одно и то же событие, их исполнение на данном участке стартует параллельно (это следует из теоремы ТМ1, учитывая, что по условию леммы данные узлы не инцидентны). Если на некоторое событие реагирует узел лишь одной из ветвей, то ее исполнение на данном участке является исключительным, что эквивалентно исполнению конструкции принятия решения.

Лемма Л3. Узел, имеющий несколько входящих дуг в узлы, инцидентные по выходам только ему, в зависимости от событийной схемы является конечной точкой условной конструкции последовательной программы и/или точкой слияния параллельных ветвей.

Доказательство. Аналогично доказательству Л2.

Теорема об интерпретации связей (ТМ2). В ходе реакции на одно событие методы объектов, соответствующих узлам, исполняются в порядке, соответствующем идеологии сетевого графика работ, если отсутствие реакции объекта на событие смоделировать реакцией, состоящей в вызове пустого или предопределенного метода.

Доказательство. При реакции на одно событие узлы модели активизируются по входам и активизируют выходы в соответствии с И-логикой, это следует из ТМ1, Л2, Л3 с учетом сделанного допущения о моделировании отсутствия реакции. Получаем логику интерпретации сетевого графика работ.

Теорема о планировании событий (ТМ3). Модель программы, трасса исполнения которой заранее неизвестна или меняется в зависимости от внешних данных, реализуется путем генерации событий и подписки на события в методах объектов-узлов модели. Начальный календарь содержит как минимум одно событие, на которое реагирует как минимум один объект.

Каждый объект А имеет возможность включить в календарь новое событие. Планирование события является атомарной транзакцией.

Каждый объект А может подписать любой метод любого иного объекта В либо на любое из еще не произошедших событий, либо на текущее, но тогда и только тогда, когда объект В следует за А. Подписка – атомарная операция по отношению к В.

На любое событие объект реагирует только в одном методе. Это либо метод, указанный при проектировании модели, либо последний из указанных по подписке, либо пустой/предопределенный метод.

Доказательство. Программируемый переход от узла А к узлу В в предельном случае может быть реализован путем условного планирования узлом А такого события, которое может быть в дальнейшем обработано исключительно узлом В. С этой целью фактически вводятся понятия подписки на событие по умолчанию и программируемой подписки. Чтобы модель программы начала работу, необходимо как минимум одно начальное событие, которое обрабатывается хотя бы одним узлом. Атомарность операции планирования события является необходимым условием отсутствия коллизий в ходе исполнения параллельного процесса.

Следствие. Если доступ к календарю открыт из любого объекта, календарь должен относиться к глобальным данным.

Теорема ТМ4. Событие удаляется из календаря сразу же после его наступления.

Теорема будет доказываться совместно с теоремой ТП1.

Утверждение У3. Объектно-событийная модель процесса/программы/алгоритма должна удовлетворять определениям, данным теоремами ТМ1, ТМ2, ТМ3, ТМ4.

Теорема о самодостаточности ОСМ (ТМ5). Объектно-событийная модель является самодостаточной в смысле реализации.

Доказательство. Любой метод любого класса модели является неким алгоритмом, который может быть реализован предельной (атомарной) или непредельной (декомпозируемой) объектно-событийной моделью, вложенной в общую.

Теорема об абстрактной предельной ОСМ (ТП1). Предельной абстрактной (атомарной) объектно-событийной моделью назовем модель, включающую единственный узел и обладающую календарем, вмещающим не более одного события. Такая модель равномощна машине Тьюринга (МТ).

Доказательство. Определим соответствие. Множество команд вида $q_i a_i \rightarrow q_j a_j d_j$ любой программы (q_i и q_j – исходное и следующее состояние, a_i и a_j – символы алфавита в текущей ячейке ленты до и после исполнения команды, d_j – направление перемещения головки), исполняемой МТ, разобьем на группы относительно q_i. Если рассматривать переход в q_i как событие с идентификатором q_i в текущем контексте, то указанные группы команд являются методами некоего класса, реагирующими на такое событие выполнением элементарных действий по замене a_i на a_j в текущей ячейке, планированию нового события с идентификатором q_j и перемещению головки в направлении d_j. Лента МТ рассматривается как глобальные данные, положение головки – внутреннее состояние (поле класса). МТ всегда обладает начальным состоянием, эквивалентным начальному событию в календаре. Все требования У3 для МТ удовлетворены.

Поскольку идентификация событий в календаре должна быть уникальной, а идентификаторами являются неоднократно встречающиеся состояния МТ, будем хранить в календаре только одно следующее событие. Этого можно добиться, удаляя из календаря текущее событие сразу же после его наступления.

Таким образом, доказана не только текущая теорема, но и теорема ТМ4.

Теорема о реальной предельной ОСМ (ТП2). Предельной реальной (неатомарной) ОСМ назовем процедуру с планированием повторного входа [0]. Это модель, состоящая из одного узла-объекта, обладающего единственным методом реакции на все события. Планирование события ограничивается планированием в начало и конец календаря. Такая модель в пределе позволяет реализовать любой алгоритм.

Доказательство. Описательные возможности такой ОСМ покрывают все возможности предельной атомарной ОСМ, эквивалентной машине Тьюринга. Следовательно, предельная неатомарная ОСМ также позволяет описать любой алгоритм. В такой одноузловой системе календарь событий может являться атрибутом единственного метода обработки. Целесообразно рассматривать его как обычную процедуру. Календарь отобразим в план этапов обработки данных. Процедура содержит серию команд, реализующих этапы обработки (реакцию на события), в том числе планирование новых этапов в начало или в конец плана. Согласно ТМ3 в начальном календаре должно присутствовать хотя бы одно событие, следовательно процедура обязана иметь по меньшей мере один начальный этап обработки в плане. Каждому этапу (вызову процедуры) соответствует комплекс значений фиксированного множества элементов данных – параметров процедуры. Такого рода процедуры назовем *процедурами с планированием повторного входа*.

Следствие. Процедура с планированием повторного входа способна реализовать последовательный алгоритм, декомпозируемый в серию схожих или идентичных по структуре и алгоритму решения подзадач с динамическим планированием последовательности их решения в соответствии со стратегией стека, дека или очереди.

Теорема о последовательных и параллельных процессах (ТПР1). В рамках объектно-событийной модели некоторой программы любой последовательный или параллельный процесс может быть порожден планированием события и описан

фрагментом модели, в котором определена реакция на данное событие. Для описания параллельных процессов необходимо представить их порождение, слияние, синхронизацию, взаимное исключение и обмен данными.

Доказательство.

1. Реализуемость и механизмы *порождения и слияния* параллельных ветвей следуют из Л2, Л3. Целесообразно планировать соответствующие события порождения A1 и слияния A2. Порождаемые процессы планируют прочие события, необходимые для описания логики их работы, на промежутке $]A1,A2[$.

Точки барьерной синхронизации можно смоделировать m узлами, каждому из которых инцидентны по входам и по выходам дуги k параллельных ветвей, $1 \le m \le k$.

Взаимное исключение k параллельных ветвей (механизм критической секции) может быть смоделировано, например, разрывом точки синхронизации (при m = 1) на два узла – слияния и порождения, между которыми следует фрагмент модели, реализующий алгоритм в пределах критической секции. Планируются необходимые события слияния A1 и порождения A2. Каждая из сливаемых ветвей планирует одно событие прохождения критической секции на промежутке $]A1,A2[$. Такая схема гарантирует требуемую логику работы.

Обмен данными между процессами может быть реализован через глобальные данные с использованием взаимного исключения ветвей при доступе.

2. Произвольный последовательный процесс может быть запущен планированием нового события, обрабатываемого или отдельным узлом, методы которого описываются некоторыми, например, предельными ОСМ, реализующими произвольные алгоритмы согласно ТП1, или комплексом таких узлов, не порождающих параллельных по событиям ветвей. Согласно Л2 и Л3 в таком комплексе можно описать условное исполнение. Согласно ТМ3 можно представить переход, в том числе замыкающий цикл.

Утверждение У4. Текст программы также может являться следствием работы иной программы (системы порождения). Его отдельные фрагменты имеют однозначное соответствие результатам исполнения отдельных звеньев системы порождения и могут быть отражены в элементы трассы ее исполнения.

Следствие. Одному звену системы порождения программ могут соответствовать различные фрагменты генерируемой программы.

Лемма Л4. Фрагменты программы, решающей задачу, генерируемые одним и тем же звеном, являются алгоритмической реализацией некоей сущности (объекта/действия/отношения).

Доказательство. Звено работает по заданному алгоритму или совокупности алгоритмов, которые согласно Л1 представляют собой методы некоего класса. Класс по определению инкапсулирует данные и методы обработки, характерные для некоего класса объектов реального мира или действия или отношения. Эти сущности могут явно присутствовать в постановке задачи либо логически выводятся из нее.

Следствие. В таком контексте события эквивалентны этапам генерации кода или, что более правомерно, этапам жизненного цикла/проявлений указанной сущности.

Теорема о порождении программ (ТМ6). Среда порождения программ может использовать объектно-событийные модели для описания и программирования решения некоторой задачи [0]. Объекты–узлы модели являются экземплярами классов предметной области задачи и включаются в модель, либо исходя из постановки задачи, сформулированной в терминах предметной области, либо исходя из логических следствий из этой постановки в соответствии с некоторой системой знаний о предметной области. Модель способна породить решающую задачу программу.

Доказательство. Следует из У4, ТМ1, Л4.

Сформулированы основные положения теории ОСМ. Доказан ряд утверждений и теорем, в том числе о предельных ОСМ. Даны теоретические выкладки, обосновывающие приме-

нение ОСМ и производных от них формализмов в программировании [0], порождении [0] и моделировании обычных и параллельных программ.

Список литературы

1. Пекунов В.В. Процедуры с планированием повторного входа в языках высокого уровня при традиционном и параллельном программировании // Информационные технологии.– 2009. – № 8. – С. 63–67.

2. Пекунов В.В. Автоматизация параллельного программирования при моделировании многофазных сред. Оптимальное распараллеливание // Автоматика и телемеханика.– 2008. – № 7. – С. 170–180.

А.А. Пепеляев

Пермский государственный технический университет

МОДЕЛИРОВАНИЕ ВЗРЫВА БЫТОВОГО ГАЗА В КИРПИЧНОМ ЗДАНИИ

Аварии зданий, вызванные взрывами бытового газа, происходят регулярно. Отказы отдельных элементов конструкций зачастую способны спровоцировать прогрессирующее разрушение здания. Данная проблема в основном затрагивает существующие газифицированные здания. Такие ситуации, как несанкционированное подключение к системе газоснабжения, халатность при пользовании газом и газовыми приборами в быту, не представляется возможным контролировать или регулировать их предотвращение.

При возникновении подобного рода ситуации внутри помещений происходит дефлаграционный взрыв – быстрое горение газовоздушной смеси, концентрация горючего в которой находится между нижним и верхним концентрационными пределами воспламенения. Взрывоопасное облако формируется