УДК 681.3.06

МОДЕЛИРОВАНИЕ ТРАНЗАКЦИЙ НА АРХИТЕКТУРНОМ УРОВНЕ ПРОЕКТИРОВАНИЯ ПРОГРАММНЫХ СРЕДСТВ

К.В. Ахтырченко

Рассматривается моделирование транзакций на архитектурном уровне проектирования программных средств. Определяется транзакционная структура программного средства, предлагается способ ее описания на языке UML. Выявляется связь транзакционной структуры и других структур архитектурного уровня проектирования.

Ключевые слова: программная архитектура, моделирование программной архитектуры, структуры программных средств, транзакционные вычисления, моделирование транзакций, транзакционная структура, тип транзакционной структуры, шаблон транзакционной структуры, UML.

1. Введение. В настоящее время транзакционная обработка информации становится одним из важнейших аспектов, определяющих корректность производимых вычислений и целостность данных, а следовательно, и характеристики качества программного средства (ПС). По данным Standish Group, уже в 1996 году 57% высококритичных (mission — critical) программных средств было построено с использованием технологий менеджеров транзакций [1]. Область применения транзакционных систем охватывает практически все секторы рынка информационных технологий. Если до середины 80-х годов данное направление организации вычислений соотносилось, прежде всего, с системами управления базами данных (СУБД), то концу 90-х основной акцент в технологиях транзакционных вычислений переместился на организацию взаимодействия распределенных бизнес – объектов (business objects) [2], которые могут быть реализованы на различных языках программирования, функционировать в рамках неоднородной операционной среды, но, тем не менее, представлять целостную систему.

Большое количество и постоянно возрастающая сложность ПС, предусматривающих транзакционную обработку информации, требуют повышенного внимания к моделированию транзакций на протяжении всего процесса разработки ПС. Сегодня понятие транзакции не ограничивается только системами управления базами данных и мониторами/менеджерами транзакций. Уже на этапе построения модели проблемной области (ПрО), работы, выполняемые в рамках автоматизируемых бизнес – процессов, группируются в потоки бизнес транзакций, с которыми соотносятся критически важные, с точки зрения достижения требуемого уровня надежности, бизнес сущности (документы, модели и т.п.). Такие потоки транзакций определяют требования к организации и выполнению транзакционных вычислений в ПС и являются прообразами транзакций, выполняемых СУБД, менеджерами транзакций и т.д. Реализуемая в ПС модель транзакционных вычислений, по сути, определяет, будет ли ПС находиться в целостном состоянии и поддерживать требуемый уровень надежности. С учетом постоянного роста сложности и увеличения трудозатрат на создание ПС, это приводит к объективной потребности моделирования транзакций на самых ранних этапах разработки ПС, в том числе и в процессе моделирования его программной архитектуры (ПА).

В настоящее время существует множество методов описания транзакционного поведения ПС в контексте методов спецификации потоков работ [3]. Существуют методы спецификации структуры и поведения транзакций [3, 4], методы оценки характеристик обеспечения конкурентного доступа и уровня надежности транзакций [4]. В тоже время, отсутствуют средства описания модели транзакций ПС на архитектурном уровне, где кроме структуры, элементами которой являются транзакции (в дальнейшем – транзакционной структуры (ТС)), также имеются логическая, модульная, процессная, физическая структуры [5, 6]. Перечисленные структуры лишь в совокупности описывают ПА, являются взаимосвязанными и требуют согласованных методов их моделирования.

Сегодня одним из наиболее распространенных языков моделирования ПС, применяемых на практике, является язык UML (Unified Modeling Language) [7, 8], который имеет строго определенную семантику, описанную на базе 4-х уровневой метамодели языка, и является расширяемым, предоставляя средства введения новых элементов языка. Язык UML является сквозным[1] языком моделирования для таких процессов разработки ПС и методов проектирования ПА, как:

[1] т.е. используется на всех стадиях разработки.

— Унифицированный Процесс Разработки Программных средств (Unified Software Development Process) [9];

— Унифицированный процесс, разработанный компанией Rational (Rational Unified Process (RUP)) [10];

— Архитектурный Метод Проектирования (Architecture Based Design Method) [11].

В каждом из них предусмотрено описание логической, модульной, процессной и физической структур на языке UML. В тоже время, в них не определяется не только способ представления ТС, метод ее построения и/или анализа, но и сама структура как таковая. В связи с этим, целями данной работы являлись

— разработка ТС, обеспечивающей моделирование транзакций на архитектурном уровне проектирования ПС;

— определение связей данной структуры с другими структурами архитектурного уровня проектирования ПС;

— разработка способов описания ТС на языке UML;

— разработка метода ее построения.

Дальнейшее изложение материала осуществляется в соответствии со следующей структурой. В Разделе 2 на основе вводимых метамодели и модели, шаблонах и типах определяется ТС ПС. Здесь же устанавливаются связи ТС с другими структурами, в совокупности описывающими архитектуру ПС. В Разделе 3 вводится аппарат описания ТС в процессе моделирования ПА, предлагается способ представления данной структуры на языке UML. В Заключении определяется место моделирования ТС в процессе разработки ПС.

2. Транзакционная структура (ТС).

2.1. Метамодель и модель ТС ПС. В основу подхода к определению ТС было положено ее первоначальное представление в виде метамодели на языке UML (Диаграмма 1).

Элементами метамодели, описываемыми в виде классов, являются:

— транзакции;

— работы (действия (activities)), выполняемые в рамках транзакций;

— объекты, связываемые с транзакцией;

— связи между транзакциями, которые классифицируются на зависимости, связи порядка выполнения и организационно – структурные связи.

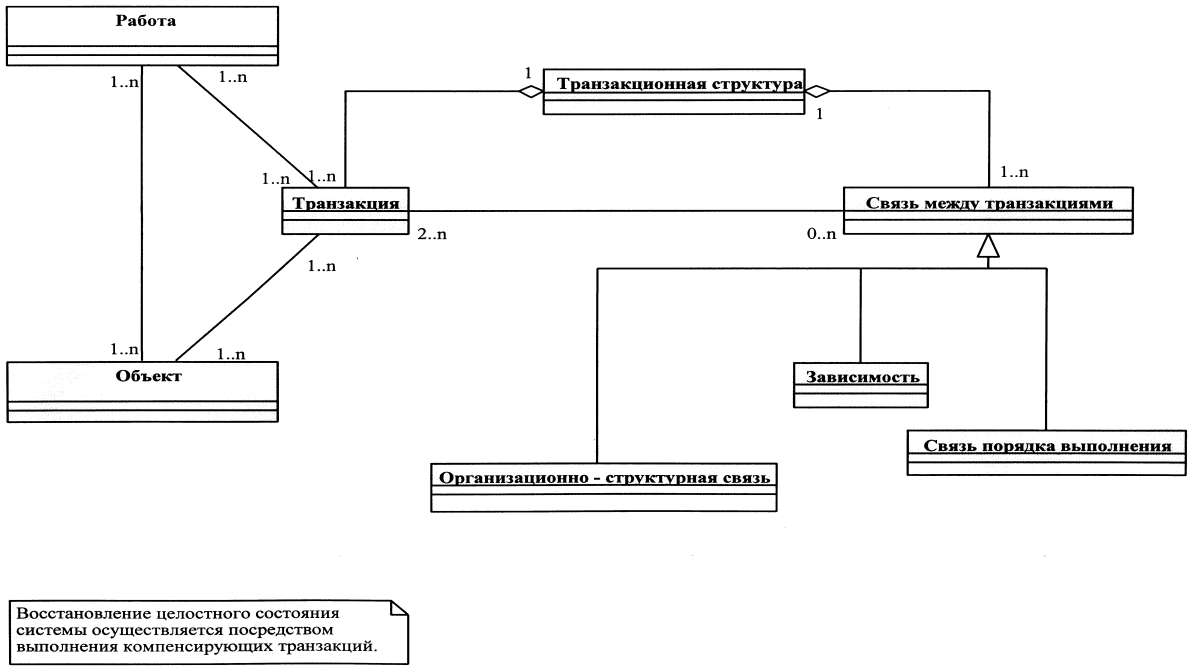

Диаграмма 1. Метамодель ТС ПС

Переход от метамодели к модели ТС осуществляется путем определения экземпляров классов метамодели, которые представляют собой типы — элементы вводимой модели ТС. Связи, устанавливаемые между классами метамодели, правомерны для определяемых экземпляров классов.

В зависимости от элементов метамодели, экземплярами классов являются следующие типы (Таблица 1).

Таблица 1

Соотношение классов — элементов метамодели и их экземпляров

п/п	**Класс – элемент метамодели**	**Экземпляр класса**
1	Транзакция	В модели ТС соответствует типу транзакций, например, “транзакция – потомок”
2	Работа	Соответствует типу работ в модели ТС
3	Объекты, связываемые с транзакцией	В модели ТС соответствует типу объектов, связываемых с транзакцией, например, “Восстанавливаемый ресурс” или “Инициатор транзакции”
4	Связь между транзакциями	Соответствует в модели ТС типу связей, устанавливаемых между транзакциями
5	Зависимость	В модели ТС соответствует типу зависимостей, например, “Зависимость по данным” или “Зависимость по прерыванию”
6	Связь порядка выполнения	Соответствует в модели ТС типу связей порядка выполнения, определяющих порядок выполнения транзакций
7	Организационно – структурная связь	В модели ТС соответствует типу организационно-структурной связи, например, “Агрегация” некоторой транзакцией – родителем N транзакций потомков

При разработке метамодели не ставилась цель создания аппарата, унифицирующего все существующие к настоящему моменту модели транзакций, а поэтому, представленная метамодель допускает расширения через вводимые классы элементов и связи между ними.

Выбор используемого подхода к определению ТС обусловлен зависимостью вводимых типов от системы моделей транзакций, которую предполагается использовать в качестве базиса при моделировании ТС ПС. Действительно, в зависимости от характеристик ПрО, в одном случае, будет достаточно модели многозвенных транзакций, а в другом — потребуется применить модель кооперативных транзакций (cooperative transaction [4]). Создание унифицированной модели ТС, объединяющей как можно большее количество моделей транзакций, приводит к сложности создаваемого аппарата моделирования и его избыточности. Последнее обусловлено тем, что на практике в большинстве случаев требуются модели плоских и многозвенных транзакций, реже — модели вложенных транзакций. Потребность в использовании других моделей транзакций возникает еще реже. Поэтому, обеспечивая, с одной стороны, простоту и минимальную достаточность модели транзакционной структуры, а с другой стороны — возможность применения специализированных (“экзотических”) моделей транзакций, предполагается, что состав системы моделей транзакций может изменяться. Вследствие этого, модель ТС становится настраиваемой, предусматривая возможность изменений через вводимые/исключаемые типы транзакций, типы зависимостей и т.д. Тогда результирующая модель ТС может включать только те средства моделирования транзакций на архитектурном уровне, которые будут действительно востребованы, что в конечном итоге способствует увеличению практической значимости предлагаемого аппарата моделирования.

Модели плоских и многозвенных транзакций в большинстве случаев можно рассматривать как частные случаи модели вложенных транзакций. Поэтому, исходя из описанного подхода и не ограничивая область применимости и значимость полученных в данной работе результатов, в дальнейшем, определение

ТС, изложение способа ее описания и метода построения будет осуществляться для случая использования модели вложенных транзакций в соответствии с правилами, сформулированными Моссом [12].

2.1.1. Определение типов транзакций. Иерархия типов транзакций строится на основе механизма наследования классов, представляющих собой типы транзакций. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип транзакции". С учетом того, что восстановление целостного состояния ПС осуществляется посредством выполнения компенсирующих транзакций, иерархия типов транзакций модели ТС может быть следующей (Диаграмма 2).

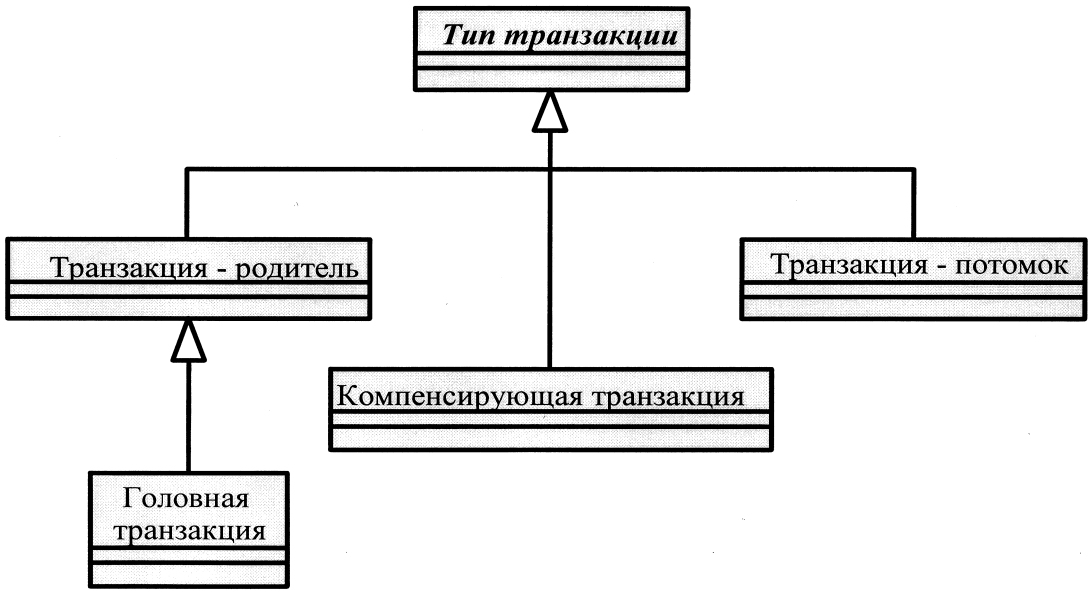

Диаграмма 2. Иерархия типов транзакций

2.1.2. Определение типов объектов, связанных с транзакцией. Также как и в случае типов транзакций, иерархия типов объектов, связанных с транзакцией (в дальнейшем – связанных объектов), строится на основе механизма наследования классов, которые представляют собой типы связанных объектов. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип связанного объекта". Типами объектов, связанных с транзакцией могут быть: ресурс, захватываемый транзакцией в процессе своего выполнения, инициатор транзакции, объект, обеспечивающий восстановление целостного состояния ПС в рамках компенсирующих транзакций (в дальнейшем – восстанавливающий объект). В случае необходимости введения явной классификации соотносимых с транзакцией ресурсов на восстанавливаемые и невосстанавливаемые, могут быть определены два класса: "Восстанавливаемый ресурс" и "Невосстанавливаемый ресурс". Они являются производными по отношению к классу "Ресурс транзакции".

2.1.3. Определение типов зависимостей. Иерархия типов зависимостей между транзакциями строится на основе механизма наследования классов, представляющих собой типы зависимостей. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип зависимости". Определение типов зависимостей является одним из важных шагов описания семантики выполнения транзакций. Для моделирования вложенных транзакций достаточной является система типов зависимостей, предложенная в [13] с учетом типов зависимостей, вытекающих из правил управления вложенными транзакциями, сформулированных Моссом [12] (Диаграмма 3).

Диаграмма 3. Типы зависимостей между транзакциями

Ниже для каждого из представленных на диаграмме типов зависимостей описывается его семантика, а для типа зависимости по данным и выполняемым в рамках транзакции действиям вводятся раскрывающие его подтипы.

Зависимости по данным и выполняемым в рамках транзакций действиям.

— Зависимость делегирования полномочий. Транзакция T1 разрешает транзакции T2 выполнять определенные действия над указанными объектами (данными).

— Зависимость ответственности за выполняемые действия. Транзакция T1 передаёт транзакции T2 ответственность за действия, выполняемые T1 над указанными объектами (данными). То есть, результаты этих действий транзакции T1 над указанными объектами будут зафиксированы в том и только в том случае, если транзакция T2 будет зафиксирована. В случае прерывания транзакции T2, результаты этих операций не будут зафиксированы в системе, даже если транзакция T1 будет зафиксирована.

— Зависимость правила видимости данных. Результаты выполнения транзакция T1 становятся видимыми транзакции T2.

— Зависимость правила видимости действий. Действия, выполненные транзакцией T1, становятся видимыми транзакции T2.

Зависимость по прерыванию. Транзакция T зависит от транзакций T1, ... , Tn по прерыванию, если в случае прерывания хотя бы одной транзакции Ti, транзакция T также должна быть прервана. Заметим, что на основании данной зависимости может быть определена зависимость, предусматривающая, что в случае прерывания транзакции T, должны быть прерваны транзакции T1, ... , Tn.

Зависимость по фиксации. Транзакция T зависит от транзакций T1, ... , Tn по фиксации, если транзакция T не может быть зафиксирована, пока не будут зафиксированы транзакции T1, ... , Tn. При прерывании какой - либо транзакции Ti, транзакция T все же может быть зафиксирована.

Групповая зависимость. Транзакции T1, ... , Tn имеют групповую зависимость по фиксации, когда либо все транзакции должны быть зафиксированы, либо, если хотя бы одна транзакция будет прервана, то и все транзакции должны быть прерваны.

2.1.4. Определение типов связей порядка выполнения. Аналогично иерархии типов зависимостей, иерархия типов связей порядка выполнения строится на основе механизма наследования классов, которые представляют собой определяемые типы. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип связи порядка выполнения". Рассматривая поток транзакций как некоторый процесс, данные связи устанавливают порядок выполнения транзакций по отношению друг к другу. Так, например, может существовать связь между транзакциями T1, T2, T3 распределенной транзакции T, определяющая, что T1 и T2 должны начать выполняться раньше, чем T3. Используемые при моделировании параллельных процессов связи порядка выполнения [14], могут быть применены и в случае выполнения транзакций.

Ниже описывается семантика вводимых типов связей.

— Связь начала выполнения. Транзакция T1 начинает выполнение до начала выполнения транзакции

Т2.

— Связь окончания выполнения. Транзакция Т1 завершает выполнение до завершения выполнения транзакции Т2.

— Связь последовательного выполнения. Транзакция Т1 завершает выполнение до начала выполнения транзакции Т2.

Важно отметить, что отсутствие связей порядка выполнения между транзакциями соответствует неопределенному порядку выполнения транзакций, который, в основном, специфичен для транзакций, обладающих свойством изолированности.

2.1.5. Определение типов организационно – структурных связей. Организационно – структурные связи, устанавливаемые между транзакциями, как правило, определяют ТС ПС с точки зрения подчиненности (вложенности) одних транзакций другим. Иерархия данных типов связей между транзакциями строится на основе механизма наследования классов, представляющих собой типы организационно – структурных связей. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип организационно – структурной связи". В большинстве случаев для моделирования бывает достаточно типа связи "Агрегация" [7, 15, 16], определяющего, в случае модели вложенных транзакций, включение одной транзакции в другую.

2.1.6. Определение типов работ. Моделирование работ не является частью процесса моделирования транзакционной структуры. В то же время, именно работы выполняются в рамках потоков транзакций, определяя тем самым характеристики транзакционных вычислений, реализуемых ПС. Примером могут быть долговременные действия, наличие которых выдвигает дополнительные требования к ПС и может привести к необходимости использования долговременных транзакций. Для работ, которые могут потребовать применения специфичных транзакционных механизмов, определяется возможность их классификации через вводимые для них типы. Иерархия типов работ строится на основе механизма наследования классов. В качестве корня иерархии — абстрактного базового класса для всех остальных элементов иерархии — выступает класс "Тип работ". На Диаграмме 4, исходя из критерия времени выполнения, представлена одна из возможных классификаций типов работ.

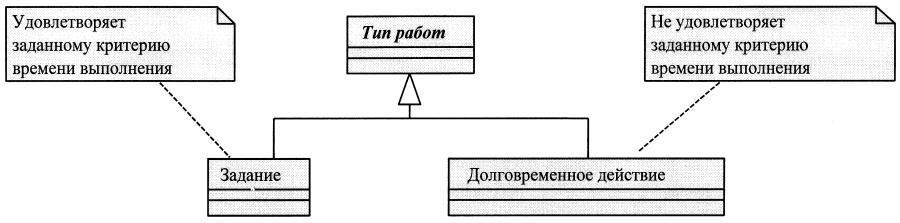

Диаграмма 4. Классификация работ, основанная на критерии времени их выполнения

Тогда, если определен некоторый интервал времени Dt, то работы, время выполнения которых не более Dt имеют тип "Задание", иначе — тип "Долговременное действие".

2.1.7. Классы характеристических объектов модели ТС. При определении модели ТС также могут быть введены классы характеристических объектов, позволяющие описывать характеристики экземпляров, типы которых введены на основе метамодели ТС.

Классами характеристических объектов являются:

— Класс "Состояние транзакции", определяющий состояния, в которых могут находиться транзакции определенных типов.

— Класс "Условие выполнения связи", экземпляры которого выражают условия выполнения связей между транзакциями. Это обеспечивает возможность описания альтернатив, возникающих при описании и выполнении транзакционных вычислений. Примерами являются: условие выполнения альтернативной транзакции – потомка, условие выбора компенсирующей транзакции, поддерживающей требуемую стратегию восстановления транзакции – родителя.

— Класс "Правило захвата", определяющий множество возможных стратегий захвата транзакциями ресурсов. Экземпляры данного класса характеризуют связь транзакции с ее ресурсами и могут рассматриваться как ассоциативные объекты, раскрывающие данную связь.

— Класс "Правило восстановления", который определяет множество возможных стратегий восстановления целостного состояния ПС.

— и т.п.

В случае класса "Состояние транзакции", может быть определен базовый набор состояний, в которых может находиться транзакция независимо от ее типа [4, 13]

— Инициирована, т.е. зарегистрирована в системе, но ещё не начала своего выполнения.

— Выполняется, т.е. начала своё выполнение.

— Завершена, т.е. выполнены все связанные с ней работы (действия).

— Зафиксирована, т.е. завершилась операцией commit.

— Прервана, т.е. завершилась операцией abort.

— Активна, т.е. является выполняющейся либо завершённой, но ещё не зафиксированной или не прерванной.

Исходя из перечисленных состояний, на Диаграмме 5 дается модель ЖЦ транзакции, определяющая ограничения на переходы из одного состояния в другое. Некорректным является любой переход, не предусмотренный в данной модели.

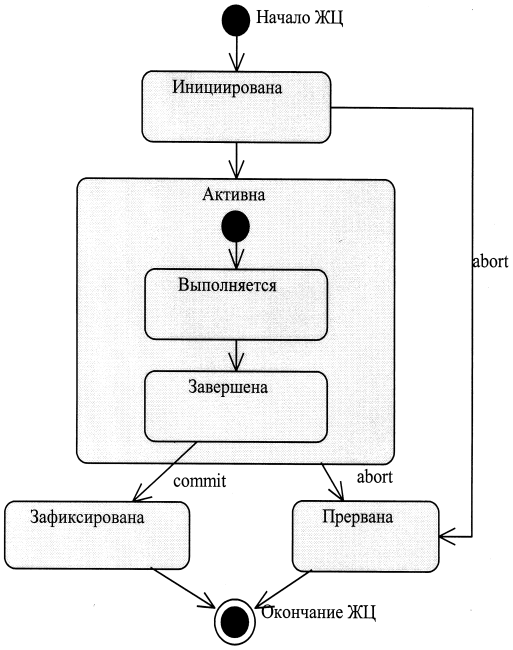

Диаграмма 5. Жизненный цикл транзакции

Вложенность состояний "Выполняется" и "Завершена" в состояние "Активна" показывает, что транзакция, являясь активной, выполняется или уже завершена.

2.2. Связь ТС с другими структурами архитектурного уровня проектирования. Моделирование ПА предусматривает построение ряда взаимосвязанных структур (чаще всего – логической (концептуальной), модульной, процессной и физической), в совокупности описывающих ПА ПС. Поэтому, для корректного определения аппарата моделирования транзакций на архитектурном уровне, одной из задач становится идентификация связей ТС с перечисленными структурами архитектурного уровня проектирования. Выявленные связи позволяют, с одной стороны, определить место и роль ТС в системе структур архитектурного уровня проектирования, а с другой стороны, осуществить интеграцию методов ее построения/анализа и существующих методов процесса разработки ПС.

2.2.1. Связь с логической структурой. Логическая (концептуальная) структура включает множество абстракций, необходимых для описания функциональных требований к системе на абстрактном уровне, т.е. уровне, не затрагивающем вопросы реализации. В случае применения объектно – ориентированных методов анализа и проектирования данная структура представляет собой объектную модель ПС, предусматривающую описание ПрО в виде совокупности взаимодействующих объектов. Поскольку в настоящее время, объектно – ориентированный подход к разработке характерен для большинства проектов построения ПС, связь между логической и транзакционной структурами устанавливается исходя из предположения, что ПС представляет собой совокупность взаимодействующих объектов. В соответствии с рассматриваемым подходом, объект характеризуется:

— множеством атрибутов, значения которых определяют состояние объекта;

— множеством методов, определяющих поведение объекта, интерфейс объекта с внешней средой, в том числе — интерфейс взаимодействия с другими объектами.

Моделирование ПС в рамках объектно – ориентированного подхода предусматривает выделение классов объектов, установление связей наследования (обобщения), агрегации между классами и др. [16]. Для описания жизненного цикла объектов, взаимодействия объектов используют [16, 17] диаграммы состояний (state diagrams), диаграммы деятельности (activity diagrams), диаграммы последовательности (sequence diagrams) и др.

При организации транзакционных вычислений, ресурсами, связываемыми с транзакциями, являются объекты. Воздействие на ресурсы осуществляется через вызовы методов объектов. Последовательность вызовов объектов определяется потоком работ, связываемым с транзакцией. В тоже время, выполнение транзакций над объектами предусматривает реализацию конкурентного доступа к объектам. Учитывая сказанное выше, становится возможной идентификация основных связей между ТС и логической структурой (см. Диаграмму 6):

— с транзакцией соотносится множество объектов, которые могут рассматриваться как разделяемые транзакциями ресурсы;

— тело транзакции определяется последовательностью вызовов методов объектов[2];

— транзакция определяет ограничения целостности, которые должны быть выполнены на уровне взаимодействующих объектов.

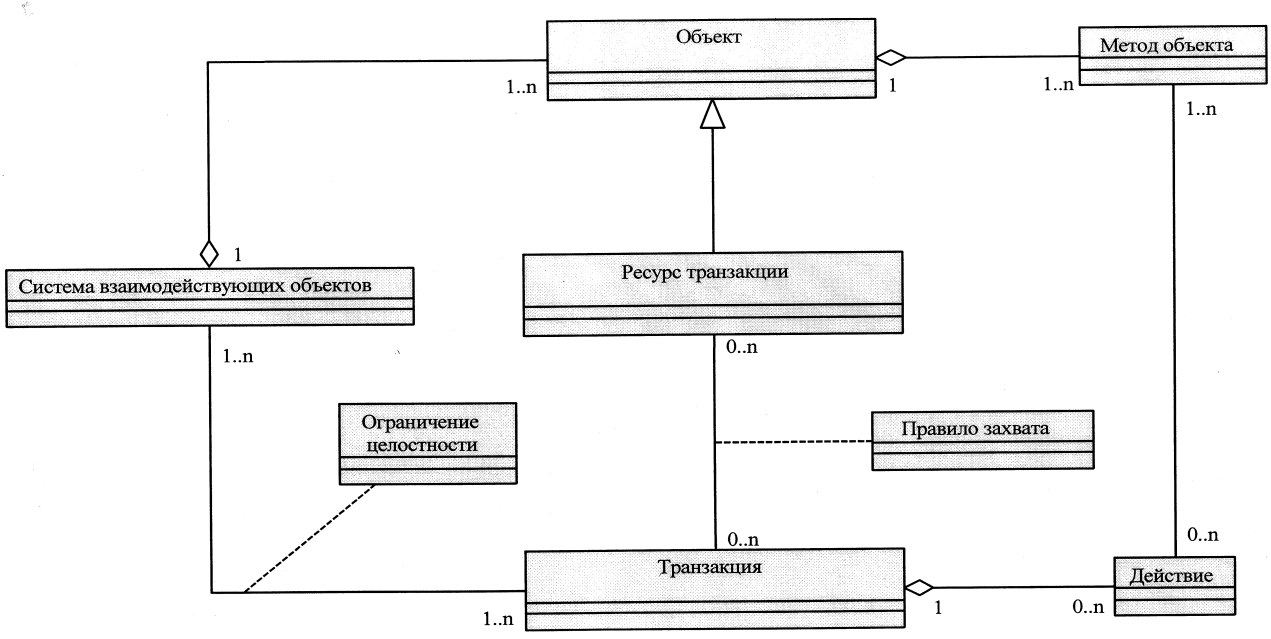

Диаграмма 6. Связь ТС и логической структуры

Важно отметить, что поток выполняющихся транзакций по отношению к объектам выступает в роли, аналогичной потоку работ, но, кроме этого, накладывает важные дополнительные ограничения на целостность производимых вычислений.

2.2.2. Связь с процессной структурой. Наиболее тесная связь возникает между процессной и транзакционной структурами ПС (см. Диаграмму 7).

[2]В случае, если у объекта не были определены методы, но определены атрибуты, т.е. объект представляет собой структуру данных, в соответствии с основополагающим принципом объектно-ориентированного программирования (инкапсуляцией объектом данных), для доступа на чтение/запись значений атрибутов по умолчанию определяются методы "получить значение атрибута" (get<NameOfAttribute>) и "установить значение атрибута" (set<NameOfAttribute>).

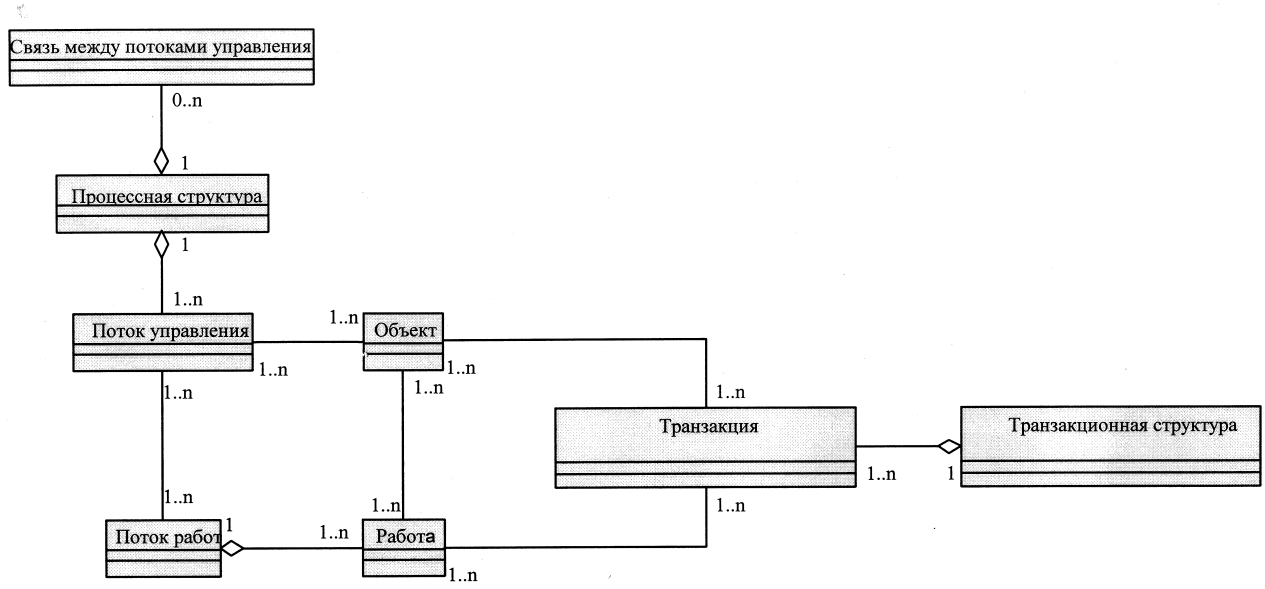

Диаграмма 7. Связь транзакционной и процессной структур

Действительно, как было сказано в предыдущем разделе, с транзакцией соотносится множество объектов. Транзакция также предусматривает выполнение действий (работ) связываемых с некоторым потоком работ. Потоки работ, реализуемые ПС, определяют потоки управления (процессы, нити (threads)). Между потоками управления и объектами устанавливаются связи принадлежности, определяющие факт функционирования объекта в рамках рассматриваемого потока управления. Учитывая, что элементами процессной структуры являются потоки управления и связываемые с ними объекты, транзакция, по сути, формирует контекст выполнения потока работ в рамках некоторого потока управления, обеспечивая целостность производимых вычислений. При этом связи между транзакциями характеризуют связи между потоками управления.

В тоже время, нельзя не отметить, что ТС определяет иной взгляд на ПС, чем его процессная структура. Это обусловлено тем, что, во-первых, элементы ТС – транзакции имеют природу, отличную от потоков управления, а во-вторых, ТС позволяет осуществить моделирование структурных аспектов организации ПС, которые не могут быть представлены на уровне процессной структуры (например, ограничения целостности, логика восстановления системы в целостное состояние). ТС может быть построена на основе процессной, но, одновременно с этим, логика выполнения транзакций может повлиять на декомпозицию работ по потокам управления.

2.2.3. Связь с модульной структурой. Связь ТС и модульной структуры реализуется через связываемые с транзакцией объекты, а также через выполняемые в рамках нее работы. Транзакция обеспечивает переход ПС из одного целостного состояния в другое. При этом выполнение транзакции сопровождается вызовом определенных методов объектов, связываемых с данной транзакцией. Разработку (проектирование и реализацию) вызываемых методов объектов в некоторых случаях можно рассматривать как единицы работы, достаточные для тестирования транзакции. Это, в свою очередь, может оказать влияние на формирование правильной модульной структуры, которая выступает в качестве базиса при распределении работ в проектной команде.

2.2.4. Связь с физической структурой. Физическая структура определяет отображение элементов ПС на аппаратное обеспечение, описывает среду эксплуатации программного средства и т.п. [5, 6]. С точки зрения транзакционной обработки информации, физическая структура показывает, как с учетом системного программного обеспечения (в том числе СУБД и средств промежуточного программного обеспечения) осуществляется развертывание компонентов ПС, реализующих логику транзакционных вычислений, по элементам физической структуры.

2.2.5. Область применения ТС. Как и в случае процессной и физической структур [5, 6], ТС имеет определенную область применения. Существуют случаи, когда при моделировании ПА использование транзакционной структуры может быть нецелесообразно. Примером может служить ПС, которое будет использоваться в однопользовательском режиме, и, при этом, не выдвигается содержательных требований

к его надежности. В целом, можно считать, что наибольший положительный эффект от моделирования ТС будет получен в случае ПС, где:

— участниками транзакций становятся бизнес - объекты, например, при построении систем в трехзвенной архитектуре;

— осуществляется создание распределенной, возможно неоднородной, базы данных.

2.3. Переход от модели ТС к ТС ПС. В качестве следующего шага определения ТС рассматривается переход от модели ТС к ТС ПС, который базируется на подходе, предусматривающем разработку шаблонов и типов ТС.

Модель ТС включает типы транзакций, связей между транзакциями и т.п. Соотносимые с моделью ТС типы обеспечивают гибкость и минимальную достаточность аппарата моделирования, что требуется для его эффективного применения на практике. Учитывая это, в основу подхода к описанию был также положен принцип, предусматривающий первоначальное описание типа ТС, в том числе на базе вводимых шаблонов. Описание типа ТС предполагает описание классов характеристических объектов и типов, вводимых на уровне модели ТС. Тогда построение ТС разрабатываемого ПС может рассматриваться как определение экземпляра требуемого типа ТС. Вводимая типизация, кроме описанных преимуществ, способствует повторному использованию архитектурных решений. Кроме того, шаблоны и типы могут разрабатываться специалистами в области транзакционной обработки информации, что увеличивает вероятность достижения требуемого для системы типов уровня качества. На Диаграмме 8 представлены основные связи между составляющими, определяющими предлагаемый подход к описанию транзакционных структур.

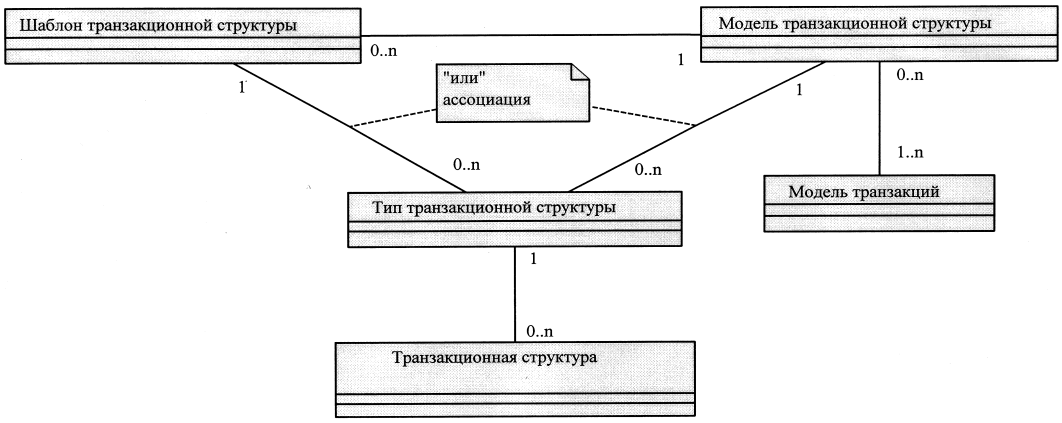

Диаграмма 8. Описание ТС через вводимые шаблоны и типы

Так, шаблоны и/или типы ТС вводятся на основе модели транзакций, с которой, в свою очередь, соотносится непустое множество моделей транзакций, образующих систему моделей транзакций. Например, такими системами могут быть:

— система 1, состоящая из модели плоских транзакций;

— система 2, состоящая из моделей плоских и вложенных транзакций;

— система 3, состоящая из модели вложенных транзакций.

Система моделей транзакций ограничивает множество возможных шаблонов и/или типов. При этом, различные системы моделей транзакций могут порождать как различные, так и одинаковые множества шаблонов и/или типов. Действительно, поскольку модель плоских транзакций может рассматриваться как частный случай модели вложенных транзакций, но не наоборот [12], система 1 и система 3 порождают различные множества возможных шаблонов и/или типов, а система 2 и система 3 — одинаковые множества.

В то время как тип ТС определяется классами характеристических объектов и типами, вводимыми на уровне модели ТС, шаблон представляет собой тип ТС, определенный с точностью до некоторых параметров. Такими параметрами могут быть:

— иерархия типов, вводимая на уровне модели ТС и предусматривающая доопределение детализирующих иерархию типов;

— количественные параметры, например, число транзакций – потомков.

Переход от шаблона к типу осуществляется посредством указания соответствующих значений

параметров. Важно отметить, что описание ТС предусматривает введение новых типов/шаблонов на основе уже существующих типов/шаблонов.

И наконец, на основании имеющегося описания шаблонов и типов, определенных классов характеристических объектов может быть описана ТС, которая представляет собой совокупность экземпляров типов транзакций, экземпляров типов связей, устанавливаемых между конкретными транзакциями, экземпляров типов работ и т.д. Примером могут служить глобальная транзакция T, набор локальных транзакций T1, ... , T3, связываемые с локальными транзакциями ресурсы – базы данных DB1, DB2, а также конкретные работы W11, ... ,W14, W21, ... ,W23, W31, ... ,W32, выполняемые в рамках соответствующих транзакций.

3. Описание ТС на UML. Одна из задач, которая должна быть решена в целях обеспечения возможности моделирования ТС, заключается в разработке средств описания ее элементов и связей, в том числе представления с помощью языков моделирования, стандартизируемых и используемых на практике при разработке ПС. Решением вышеуказанной задачи является разработанный и представленный в данном разделе способ описания ТС на языке UML.

Применительно к данной работе, для целей представления элементов и связей ТС язык UML был расширен через вводимые стереотипы и переопределение семантики действий (activity) в случае их использования для представления транзакций.

Описание на языке UML ТС в целом предусматривает наличие способов представления:

— элементов модели ТС, в том числе – типов транзакций, типов связей, классов характеристических объектов и др.;

— шаблонов ТС;

— типов ТС;

— ТС, как совокупности экземпляров типов, вводимых на уровне модели ТС;

— связей с другими структурами архитектурного уровня проектирования.

Требуемые способы представления определяются посредством:

— перечисления диаграмм языка UML, определяющих способ представления;

— перечисления стандартных элементов и связей языка UML, используемых для представления ТС;

— вводимых стереотипов описания и/или ограничений;

— переопределения семантики элементов языка UML применительно к описанию транзакций.

3.1. Описание модели ТС.

Абстрактные классы

При описании модели ТС могут быть определены абстрактные классы, которые выступают в качестве корней иерархий типов связей, а также применяются при описании классов характеристических объектов. Для их описания используются абстрактные классы языка UML.

Важно отметить, что абстрактные классы могут быть введены не только для типов связей, но и для других элементов модели ТС.

Типы транзакций.

Иерархии типов транзакций на языке UML описываются в виде диаграммы классов, где в качестве классов выступают типы транзакций. Вводимый стереотип “transaction type” явным образом указывает, что представленный на диаграмме класс представляет собой тип транзакций. Между классами могут устанавливаться связи обобщения (наследования), определяющие факт наследования свойств базовых типов производными типами. Для типов транзакций допускается использование множественного наследования.

Типы связанных объектов.

Для описания типов объектов, связанных с транзакцией (типов связанных объектов), используется диаграмма классов. Тип связанного объекта представляется в виде класса, для которого определяется стереотип “constrained object type”. Между классами могут быть установлены связи наследования, определяющие иерархию типов связанных объектов.

Типы зависимостей.

Так же, как и в случае типов транзакций и типов связанных объектов, типы зависимостей описываются на UML в виде классов, но с указанием стереотипа “transaction dependency type”. Между классами могут быть определены связи наследования. Дополнительно, при описании типов зависимостей определяется семантика, соотносимая с каждым из типов. Поскольку в языке UML изначально не было предусмотрено описание семантики типов зависимостей, для ее представления предлагается использовать примечания (notes), в которых дается либо непосредственное описание семантики (например, на естественном языке или через систему примитивов, вводимую в [13]), либо указывается ссылка на

документ, описывающий требуемую семантику.

Типы связей порядка выполнения.

Типы связей порядка выполнения описываются на UML аналогично типам зависимостей, но с указанием для каждого вводимого класса стереотипа “transaction order type”.

Типы организационно – структурной связей.

Описываются аналогично другим типам связей, но с указанием для вводимых классов стереотипа “transaction organization type”.

Типы работ.

Типы работ на языке UML представляются в виде классов со стереотипами “activity type”. Между классами могут быть определены связи наследования.

Классы характеристических объектов.

Для описания классов характеристических объектов используются диаграммы классов, на которых отображаются сами классы объектов, связи между ними, а также связи классов и других элементов модели ТС. На диаграммах классы характеристических объектов указываются со стереотипом “characteristic object type”.

3.2. Описание типа ТС. Описание на UML типа ТС, определяемого на основе некоторой модели ТС, предусматривает описание следующих связей:

— ассоциаций с работами (заданиями);

— ассоциаций со связанными объектами;

— организационно – структурных связей;

— зависимостей;

— связей порядка выполнения.

Способы их описания на языке UML представлены в данном разделе. Заметим, что при описании типа ТС могут быть доопределены типы, детализирующие вводимые на уровне модели ТС иерархии.

Ассоциации с работами (заданиями).

Ассоциации с работами раскрывают связи типов транзакций и типов работ, показывая тем самым:

— работы каких типов могут быть выполнены в рамках транзакций определенного типа;

— количество работ определенного типа (множественность работ), которые могут быть выполнены в рамках экземпляра транзакции определенного типа;

— количество транзакций определенного типа (множественность транзакций), с которыми может быть связана работа определенного типа.

Представление данных связей на языке UML обеспечивается через вводимые ассоциации между классами объектов (см. Диаграмму 9).

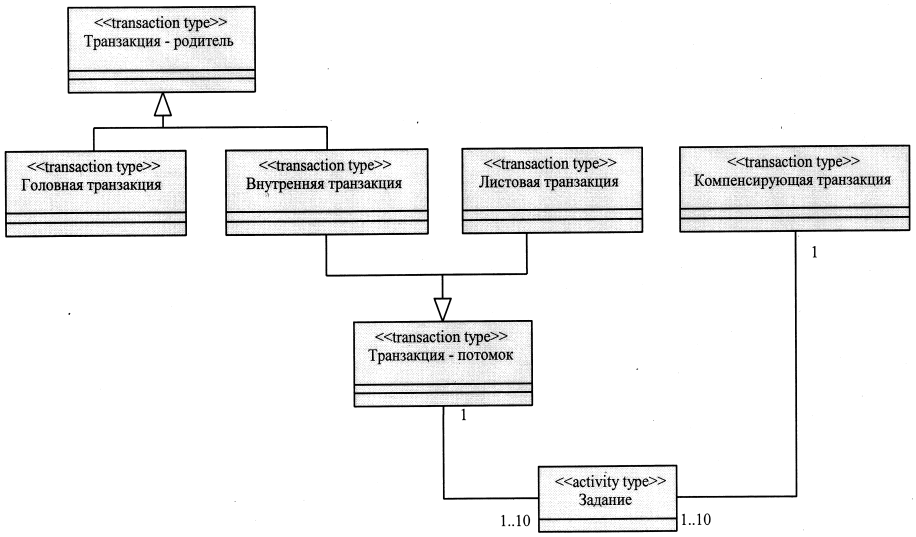

Диаграмма 9. Представление на языке UML ассоциаций с работами

Ассоциации со связанными объектами.

В отличие от ассоциаций с работами, ассоциации со связанными объектами раскрывают связи типов транзакций и типов связанных объектов, показывая тем самым:

— связанные объекты каких типов соотносятся с экземпляром транзакции определенного типа;

— количество связанных объектов определенного типа (множественность связанных объектов), которые могут быть связаны с экземпляром транзакции определенного типа;

— количество транзакций определенного типа (множественность транзакций), с которыми может быть связан экземпляр определенного типа связанных объектов.

Как и в предыдущем случае, представление данных связей на языке UML обеспечивается через вводимые ассоциации между классами объектов (см. Диаграмму 10).

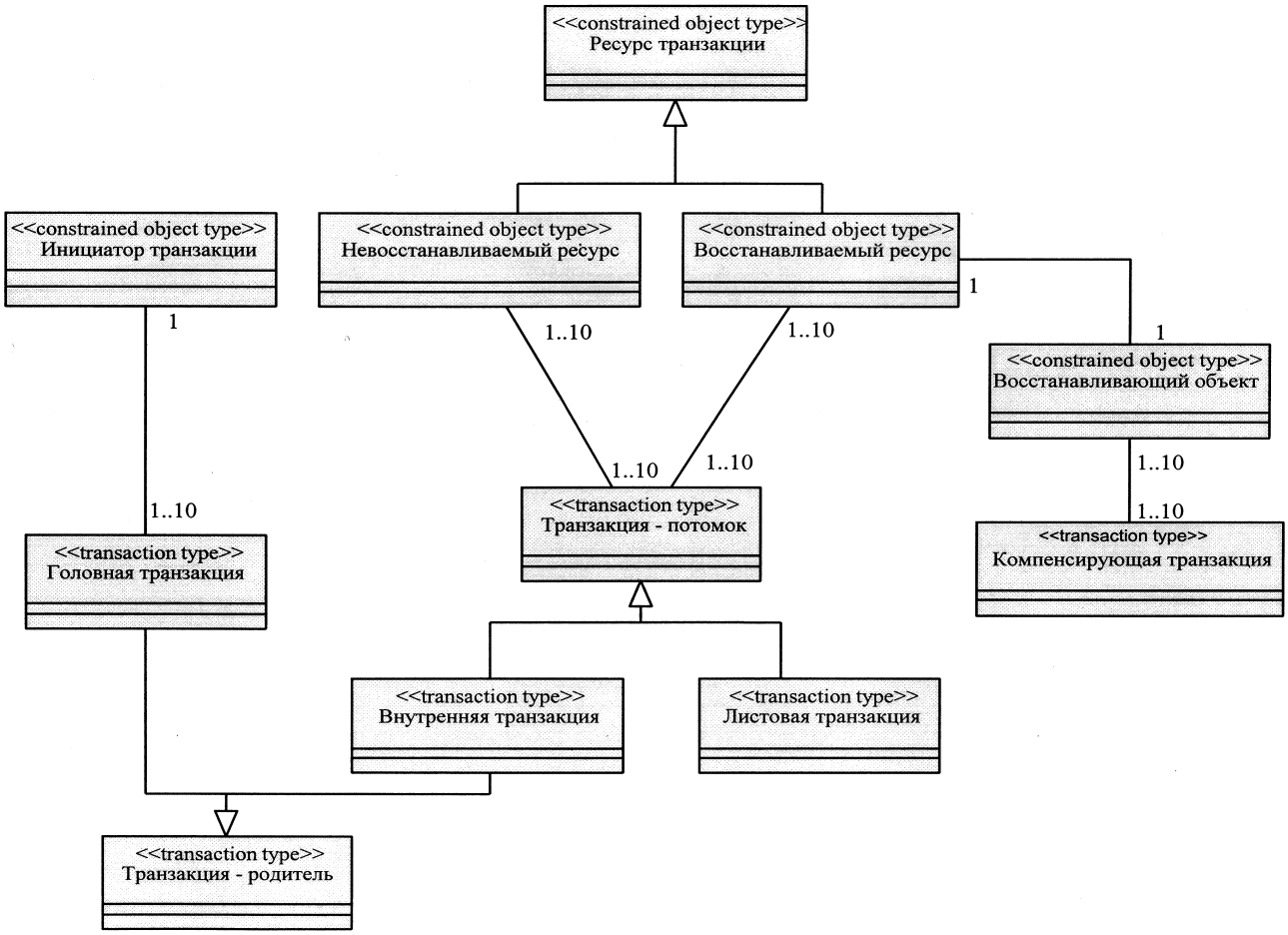

Диаграмма 10. Представление на языке UML ассоциаций со связанными объектами

Организационно – структурные связи.

Описание типа ТС предусматривает определение организационно – структурных связей между типами транзакций. Данные связи показывают:

— вложенность (агрегацию) транзакций определенных типов по отношению к транзакции некоторого типа. Другими словами, на уровне типов транзакций устанавливаются связи “часть - целое”;

— количество транзакций определенного типа (множественность агрегируемых транзакций), которые могут быть агрегированы транзакцией соответствующего типа;

— количество транзакций определенного типа (множественность агрегирующих транзакций), которыми может быть агрегирована транзакция соответствующего типа.

Представление данных связей на языке UML обеспечивается на основе вводимых при определении модели ТС типов связей. Их экземпляры устанавливаются между классами, представляющими типы транзакций, и на языке UML соответствуют связи “Агрегация” (см. Диаграмму 11).

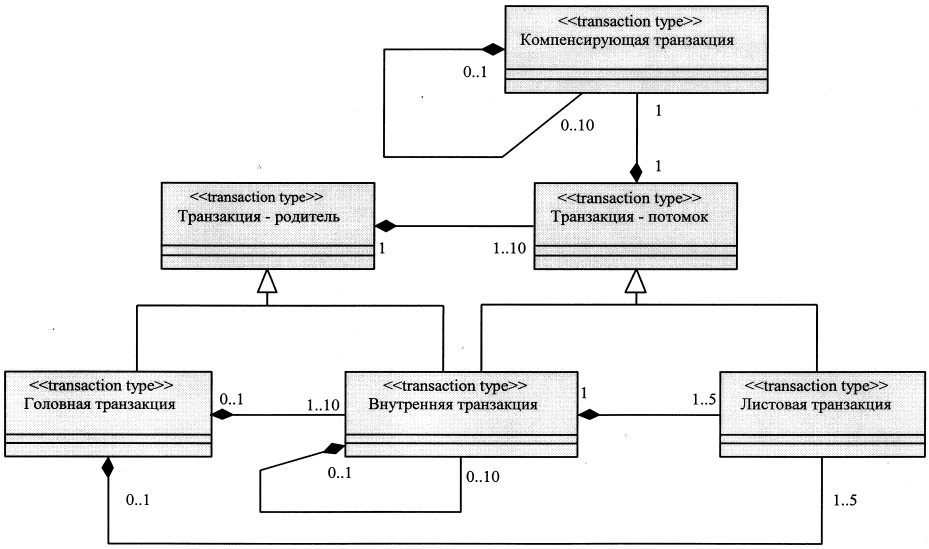

Диаграмма 11. Представление на языке UML организационно – структурных связей

Заметим, что в соответствии с семантикой языка UML, любые связи, устанавливаемые между базовыми классами, распространяются на производные классы.

Связи порядка выполнения.

При описании типа ТС также определяются связи порядка выполнения, которые могут быть идентифицированы на уровне типов транзакций. Описание данных связей на языке UML осуществляется в контексте выявленных организационно – структурных связей. Для каждого типа связей порядка выполнения определяется стереотип, обеспечивающий представление связи на диаграммах языка UML. Описание связи реализуется через указание следующей информации, которая помещается в примечание (note), соотносимое с определенной организационно – структурной связью (см. Диаграмму 12):

— направление связи, например, “Транзакция – потомок” to “Транзакция – родитель”;

— название стереотипа, соответствующего устанавливаемой связи, например, “begin earlier”.

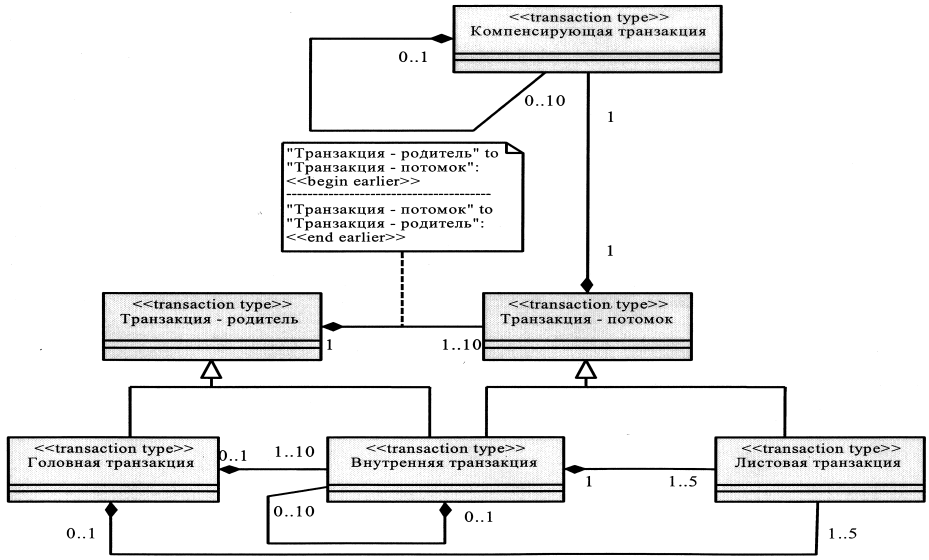

Диаграмма 12. Представление на языке UML связей порядка выполнения

Для типов связей порядка выполнения, определенных в описанной ранее модели ТС могут использоваться следующие стереотипы:

— “Связь начала выполнения” — стереотип “begin earlier”;

— “Связь окончания выполнения” — стереотип “end earlier”;

— “Связь последовательного выполнения” — стереотип “sequential order”.

Зависимости.

И наконец, при описании типа ТС определяются зависимости, которые могут быть идентифицированы на уровне типов транзакций, т.е. являются правомерными для всех транзакций затрагиваемых зависимостью типов. Так же, как и в случае описания связей порядка выполнения, описание данных зависимостей на языке UML осуществляется в контексте выявленных организационно – структурных связей на основе вводимых стереотипов. Описание зависимости реализуется через указания следующей информации, которая помещается в примечание (note), соотносимое с определенной организационно – структурной связью (см. Диаграмму 13):

— направление зависимости, например, "Транзакция – потомок" to "Транзакция – родитель";

— название стереотипа, соответствующего устанавливаемой зависимости, например, "commit dependency".

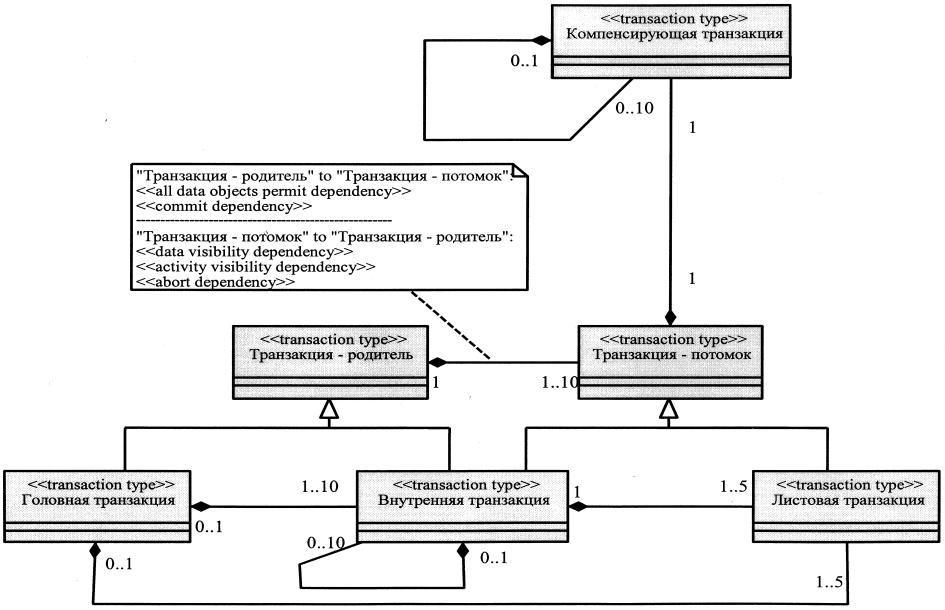

Диаграмма 13. Представление на языке UML зависимостей

Для типов зависимостей, определенных ранее в рамках модели ТС, могут использоваться следующие стереотипы:

— "Зависимость правила видимости данных" — стереотип "data visibility dependency";

— "Зависимость правила видимости действий" — стереотип "activity visibility dependency";

— "Зависимость ответственности за выполняемые действия" — стереотип "activity responsibility dependency";

— "Групповая зависимость" — стереотип "group dependency";

— "Зависимость по фиксации" — стереотип "commit dependency";

— "Зависимость по прерыванию" — стереотип "abort dependency";

— "Зависимость делегирования полномочий на все связанные с транзакцией объекты данных" — стереотип "all data object permit dependency".

3.3. Описание шаблона ТС. Шаблон ТС описывается так же, как и тип ТС, но с точностью до определенных параметров. Такими параметрами могут быть:

— иерархия типов, вводимая на уровне модели транзакционной структуры и предусматривающая доопределение детализирующих иерархию типов;

— количественные параметры, например, число транзакций – потомков.

Первый тип параметров определяется примечанием (note), соотносимым с требующей уточнение иерархией типов (см. Диаграмму 14). Примечание содержит список классов, которые требуют дальнейшей детализации через механизм наследования. Оно также может содержать перечень связей, которые в обязательном порядке должны быть определены и описаны в процессе уточнения рассматриваемой иерархии типов.

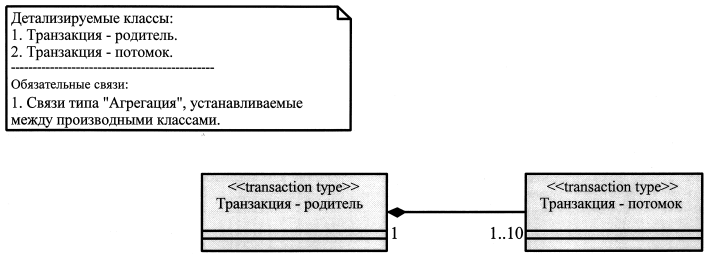

Диаграмма 14. Представление на языке UML параметров первого типа.

Второй тип параметров предусматривает выделение на UML диаграммах связей, множественность которых либо не определена вообще, либо не определена с точностью до конкретных значений (конкретного диапазона значений) (см. Диаграмму 15).

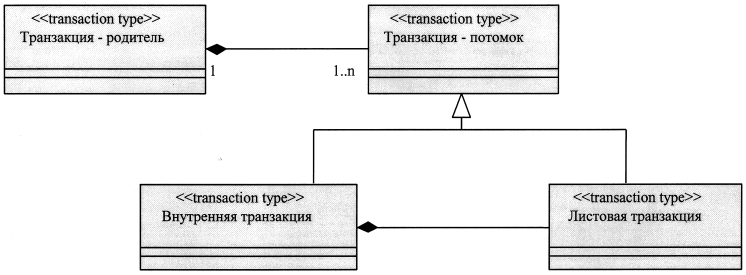

Диаграмма 15. Представление на языке UML параметров второго типа.

3.4. Описание экземпляра ТС

Описание экземпляра ТС, определяемой на основе некоторого выбранного типа ТС, предусматривает описание:

— связей с потоками работ;

— иерархии транзакций;

— связанных объектов;

— ассоциаций со связанными объектами;

— логики восстановления конфигурации транзакций в целостное состояние.

Способы их описания на языке UML представлены в соответствующих подразделах данного раздела.

Важно отметить, что экземпляр ТС, определяющий организацию транзакционных вычислений в ПС, имеет непосредственную связь с другими артефактами (моделями, документами и т.д.), создаваемыми в процессе разработки ПС. Например, иерархия транзакций вытекает из связей транзакций с потоками работ (или вычислительными процессами), реализуемыми ПС. Поскольку в настоящей работе дается разработанный способ описания ТС на языке UML, то выдвигается требование возможности моделирования на языке UML других структур архитектурного уровня проектирования, в том числе логической, модульной, процессной и физической структур, а также требований к ПС. За основу может быть взят подход к моделированию ПС, реализуемый в Унифицированном Процессе Разработки Программных средств (USDP). Тогда набор артефактов, на основании которого может быть построена конфигурация транзакций, будет включать:

— диаграммы вариантов использования (use case diagrams) и раскрывающие их диаграммы деятельности (activity diagrams);

— диаграммы классов (class diagrams), на которых явным образом выделяют классы сохраняемых в базе данных пассивных сущностей (entities) и классы активных управляющих объектов (controls);

— диаграммы деятельности, описывающие реализуемые программным средством потоки работ с учетом зон ответственности (swimlanes), определенных в языке UML;

— диаграммы классов, определяющие процессный взгляд (process view) на программное средство,

— диаграммы компонентов (component diagrams);

— диаграммы развертывания (deployment diagrams).

Связи с потоками работ.

Транзакция предусматривает выполнение внутри себя работ (действий), являющихся составной частью реализуемого ПС потока работ. Одним из основных способов описания потока работ на языке

UML является его представление в виде диаграмм деятельности (activity diagrams) [7, 8, 17]. Первичное построение данных диаграмм применительно к разработке ПС, как правило, осуществляется в процессе формирования функциональных требований к системе, когда с идентифицированными вариантами использования связываются раскрывающие их диаграммы деятельности.

При этом реализуемые ПС потоки работ описываются без учета его внутренней организации – ПС рассматривается в качестве "черного ящика". На этапе анализа и проектирования, в процессе построения объектной модели ПС, также осуществляется описание потоков работ, но уже в терминах взаимодействующих объектов. Здесь, в отличие от этапа формирования требований, на диаграммах деятельности представляются как поток управления, так и связываемый с ним поток данных. Кроме этого, определяются зоны ответственности (swimlanes), которые обеспечивают распределение ответственности за выполнение тех или иных действий внутри потока между активными управляющими объектами.

Перечисленные варианты диаграмм деятельности содержат исходную информацию, позволяющую не только определить и описать связи транзакций с потоками работ, но и, как будет видно далее, выступают в качестве базиса при построении экземпляра ТС в целом. Действительно, на основании описания потоков работ в виде диаграмм деятельности, работы могут быть распределены по транзакциям, в рамках которых они выполняются. Как результат, будут определены связи транзакций с потоком работ. Для их описания на языке UML используются диаграммы деятельности, на которых транзакции представляются в виде действий (activity) со стереотипом "transaction" (см. Диаграмму 16). Название действия, представляющего собой транзакцию (в дальнейшем транзакция), содержит название и тип транзакции, например, "Резервирование товара: Листовая транзакция".

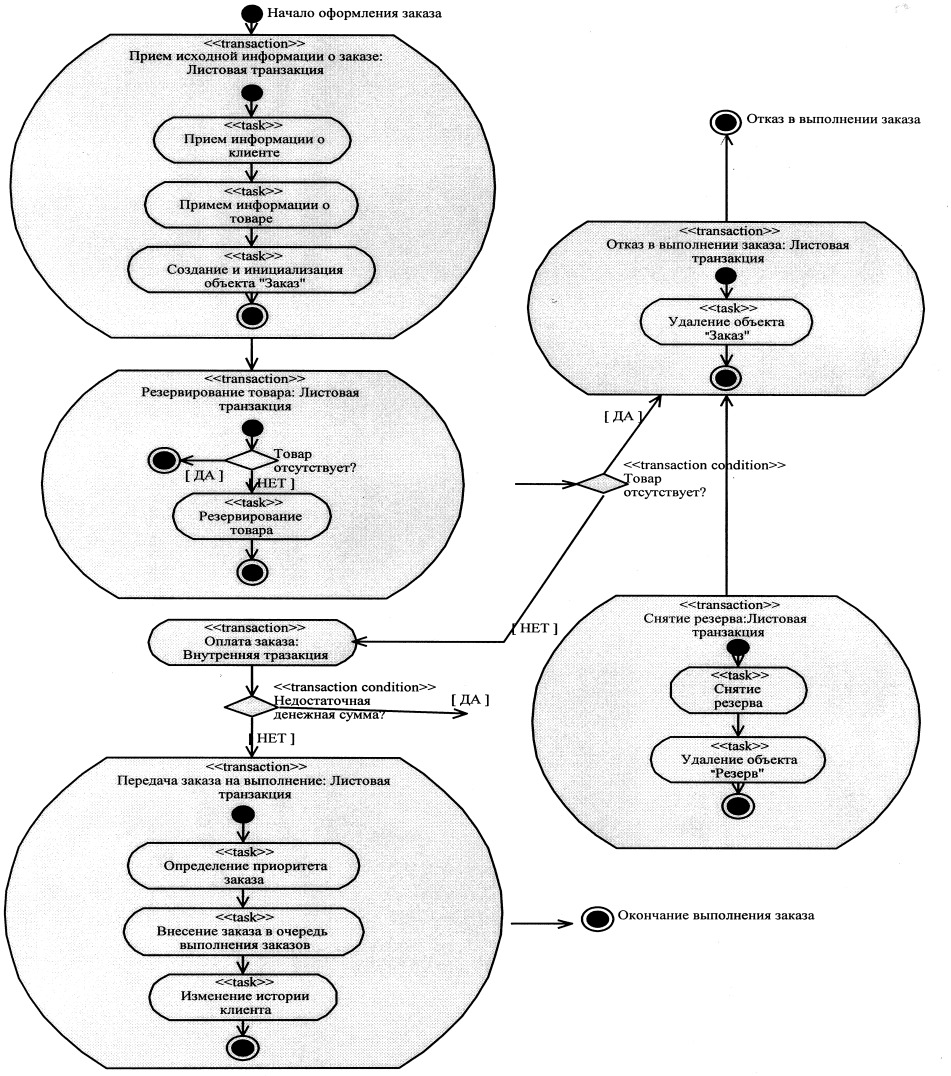

Диаграмма 16. Представление транзакций с помощью диаграмм деятельности языка UML

С транзакциями могут быть связны вложенные диаграммы деятельности, представляющие фрагменты потока работ, выполняемые в рамках транзакций. При этом действия указываются со стереотипом (на диаграмме "task"), соответствующим типу работ (на диаграмме "Задание"), определенному в используемом при построении экземпляра ТС типе ТС.

Переходы (transition) между транзакциями устанавливают порядок выполнения транзакций, который не должен противоречить связям порядка выполнения, определенным в используемом типе ТС. При этом могут быть определены условия выполнения переходов, которые соответствуют экземплярам класса характеристических объектов "Условие выполнение связи" и представляются на UML как точки принятия решения (Decision) со стереотипом "transaction condition". С точкой принятия решения связывается условие, выполнение которого определяет порядок выполнения транзакций.

Для представления вложенных транзакций используются связываемые с транзакциями – родителями вложенные диаграммы деятельности (см. Диаграмму 17).

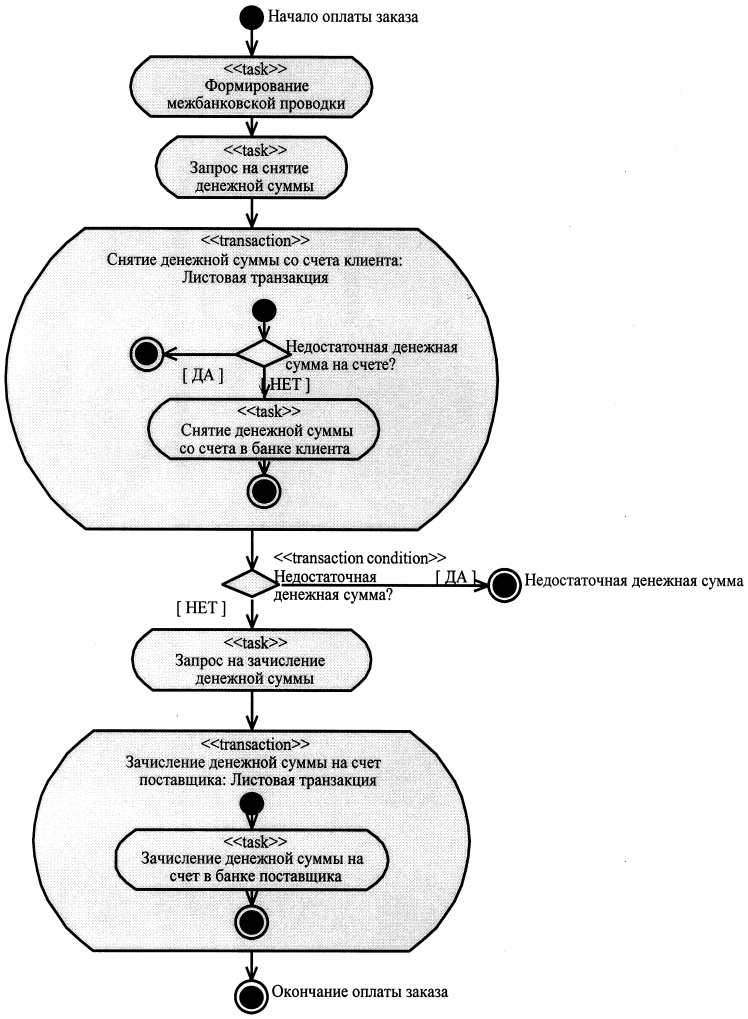

Диаграмма 17. Представление на UML вложенных транзакций (Оплата заказа: Внутренняя транзакция)

Заметим, что используемые вложенные диаграммы деятельности фактически определяют иерархию транзакций, а следовательно, должны строиться в соответствии с организационно – структурными связями, соотносимыми с используемым типом ТС.

Иерархия транзакций.

Для представления на языке UML, определяющих конкретную иерархию транзакций, организационно-структурных связей между транзакциями используются диаграммы классов (см. Диаграмму 18).

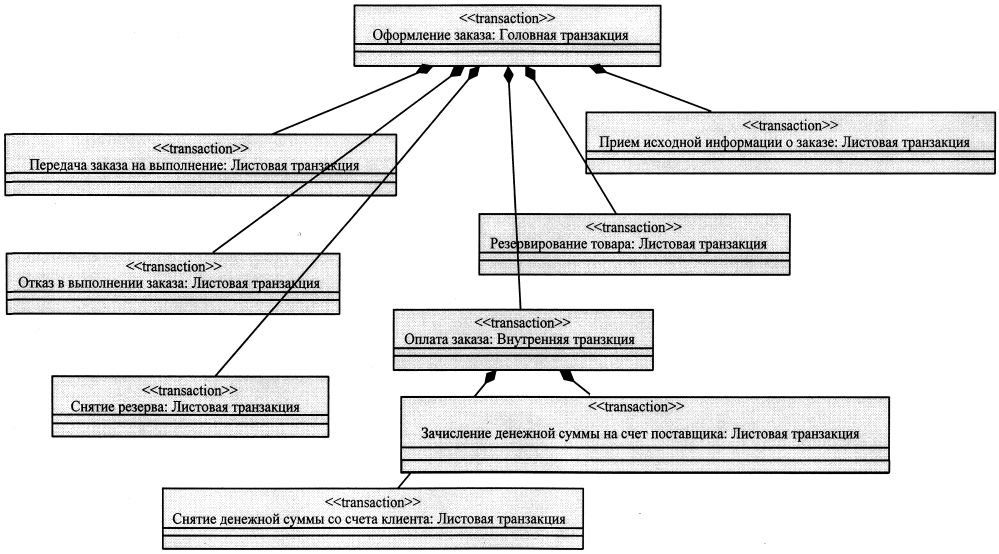

Диаграмма 18. Представление на UML иерархии транзакций

Транзакции представляются в виде классов со стереотипом “transaction”. Название класса содержит название и тип транзакции, например, “Резервирование товара: Листовая транзакция”. Организационно – структурные связи описываются с помощью ассоциаций, показывающих агрегацию одних транзакций другими. При этом устанавливаемые между транзакциями связи не должны противоречить связям, определенным в рамках используемого типа ТС.

Между транзакциями также определяются зависимости, которые не могли быть идентифицированы при определении используемого типа транзакционной структуры. Способ представления зависимостей аналогичен способу, применяемому при описании зависимостей в типах ТС, но Примечание (note) соотносится не со связями, а с транзакциями (см. Диаграмму 19).

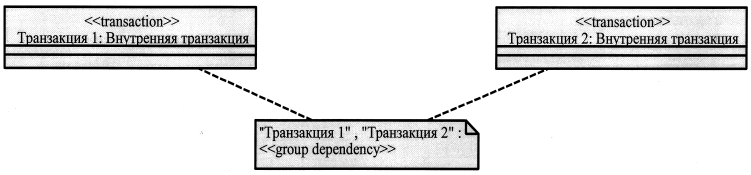

Диаграмма 19. Представление на UML зависимостей между транзакциями

Так же, как и в случае с организационно – структурными связями, вводимые на уровне экземпляра ТС зависимости не должны противоречить зависимостям, которые определены в используемом типе ТС.

Связанные объекты.

Множество связываемых с транзакциями объектов является подмножеством объединения множества пассивных сущностей и множества активных управляющих объектов, каждое из которых представляется на языке UML в виде диаграмм классов.

Результирующее множество связанных с транзакциями объектов также представляется в виде диаграммы классов, но при этом пассивные сущности и активные управляющие объекты классифицируются в соответствии с типами связанных объектов, определенных в используемом типе ТС. Для указания типа связанных объектов, название класса приобретает следующий вид “Название класса: Название типа связанного объекта”, например “Заказ: Восстанавливаемый ресурс”.

Ассоциации со связанными объектами.

Явная связь между транзакциями и связанными объектами определяется через направленные ассоциации, устанавливаемые между классами – транзакциями и классами связанных объектов. Для представления данных связей на языке UML используются диаграммы классов (см. Диаграмму 20).

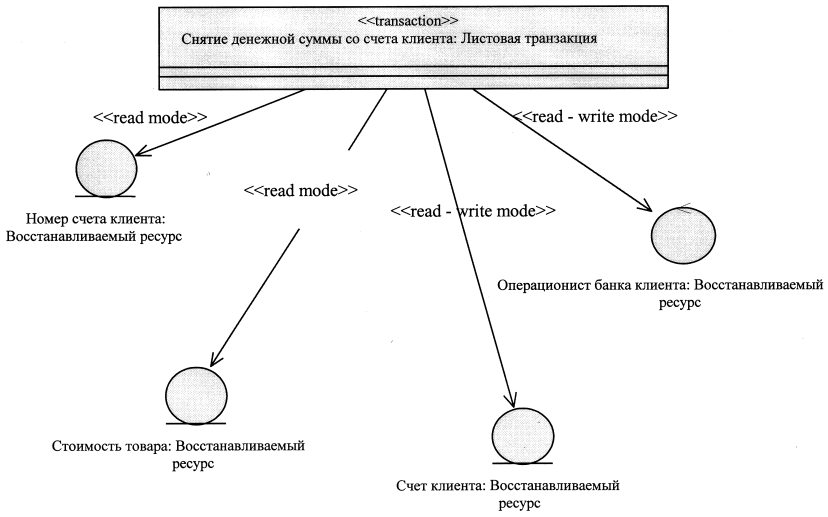

Диаграмма 20. Ассоциации со связанными объектами

Кроме того, на данной диаграмме применительно к каждой паре "транзакция : связанный объект" определяется характеристический объект "Правило захвата", который представляется в виде названия направленной ассоциации.

Логика восстановления экземпляра ТС в целостное состояние.

Многие системы управления распределенными транзакциями реализуют предопределенную стратегию восстановления экземпляра ТС в целостное состояние. С учетом того, что корректное моделирование всех аварийных ситуаций, которые могут возникнуть в системе, требует большого количества времени и ресурсов, использование стандартных стратегий восстановления можно считать приемлемым для большинства ПС, в рамках которых осуществляются транзакционные вычисления. Как результат, описание логики восстановления экземпляра ТС в целостное состояние может не осуществляться. В тех случаях, когда все же возникает необходимость в данном виде деятельности, моделирование логики восстановления может осуществляться в соответствии с подходом, описанным ниже.

Совокупность компенсирующих транзакций и связей между ними рассматривается как дополнительный экземпляр ТС, способ описания которого аналогичен способу представленному в предыдущих разделах. Отличия заключаются в том, что:

— в качестве работ (действий) выступают компенсирующие работы, обеспечивающие восстановление системы в целостное состояние;

— в качестве связанных объектов выступают восстанавливающие объекты.

Между элементами дополнительного экземпляра ТС и остальными элементами экземпляра ТС устанавливаются следующие связи:

— с каждой транзакцией – потомком связываются компенсирующие работы, которые обеспечивают ее корректное восстановление и в совокупности представляют собой компенсирующий поток работ (способ представления на языке UML – диаграммы деятельности);

— компенсирующие работы объединяются в компенсирующие транзакции, причем могут быть определены альтернативные компенсирующие транзакции (способ представления на языке UML – диаграммы деятельности);

— строится иерархии компенсирующих транзакций (способ представления на языке UML – диаграммы классов);

— на основе анализа компенсирующего потока работ и связанных с ним компенсирующих транзакций выделяются восстанавливающие объекты (способ представления на языке UML – диаграммы классов);

— с каждым восстанавливаемым ресурсом связываются восстанавливающие его объекты (способ представления на языке UML – диаграммы классов).

Дополнительно с каждой транзакцией – родителем, не являющейся компенсирующей транзакцией, может быть определено правило восстановления, описывающее семантику восстановления данной транзакции и ее транзакций – потомков.

Другие элементы и связи экземпляра ТС.

Ранее были определены способы представления на языке UML основных элементов и связей

экземпляра ТС. При этом организация транзакционных вычислений рассматривалась без учета среды реализации: конкретных операционных систем, менеджеров транзакций, вычислительных узлов и т.д. В тоже время, при разработке ПС, реализующих транзакционные вычисления, может возникнуть потребность в описании связей, характеризующих развертывание транзакций:

— по пользовательским процессам операционной системы;

— по программным компонентам;

— по вычислительным узлам.

Кроме того, функционирование ПС, реализующего транзакционные вычисления, может предусматривать наличие определенных вспомогательных компонентов (например, менеджеров транзакций), обеспечивающих управление транзакционной обработкой информации, что требует их явного представления на архитектурном уровне проектирования.

Для описания перечисленных связей, а также вспомогательных компонентов на языке UML, могут быть использованы диаграммы классов, диаграммы компонентов и диаграммы развертывания.

4. Заключение. Построение ТС ПС является деятельностью, которая может рассматриваться как составная часть процесса моделирования его ПА. Существующие процессы разработки ПС, например, Унифицированный Процесс Разработки Программных средств (USDP), предусматривают на архитектурном уровне построение логической, модульной, процессной и физической структур, связь которых с транзакционной структурой показана в разделе "Связь ТС с другими структурами архитектурного уровня проектирования". С учетом выявленных связей, ТС и метод ее построения можно рассматривать как дополнительное средство, позволяющее при моделировании ПА описать те аспекты структурной организации ПС, которые соотносятся с транзакционными вычислениями.

Важно отметить, что артефакты, описывающие ТС ПС, не имеют аналогов среди артефактов – результатов применения USDP и требуют расширения языка UML. С точки зрения последовательности выполнения работ по моделированию ПА, разработка транзакционной структуры должна выполняться после построения логической структуры ПС. В тоже время, разработка модульной, процессной и физической структур должна осуществляться уже с учетом транзакционной. При этом процессная и физическая структуры непосредственно содержат элементы и связи (например, компоненты – менеджеры транзакций), которые определяют отображение ТС на среду реализации.

СПИСОК ЛИТЕРАТУРЫ

1. *Edwards J., DeVoe D.* 3-tier client/server at work. New Work: Wiley Computer Publishing, 1997.
2. *C*ommon Business Object and Business Object Facility. OMG TC Document CF/96-01-04. http://www.omg.org
3. *Рузинкевич М., Цикоцки А.* Определение и выполнение потоков транзакций // СУБД, 1995. N 2. 106-115. N 4. 58-68.
4. *Chrysanthis P., Ramamritham K.* ACTA : A framework for specifying and reasoning about transaction structure and behavior // Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, pp. 194-203. Atlantic City, 1990.
5. *Bass L., Clements P., Kazman R.* Software architecture in practice. Reading: Addison Wesley, 1998.
6. *Clements P.C., Northrop L.M.* Software architecture: an executive overview. Technical Report CMU/SEI-96-TR-003, ESC-TR-96-003. Pittsburgh, 1996.
7. *O*MG Unified Modeling Language Specification. Version 1.3. June 1999. http://www.omg.org
8. *Booch G., Rumbaugh J., Jacobson I.* The unified modeling language user guide. Reading: Addison Wesley, 1999.
9. *Jacobson I., Booch G., Rumbaugh J.* The unified software development process. Reading: Addison Wesley, 1999.
10. *Kruchten P.* The Rational unified process: an introduction. Reading: Addison Wesley, 1999.
11. *Bachmann F., Bass L., Chastek G, Donohoe P., Peruzzi F.* The architecture based design method. Technical Report CMU/SEI-2000-TR-001, ESC-TR-2000-001. Pittsburgh, 2000.
12. *Moss J.E.B.* Nested transactions: an approach to reliable computing. Cambridge: MIT Press, 1985.
13. *Biliris A., Dar S., Gehani N., Jagadish H.V., Ramamritham K.* ASSET: A system for supporting extended transactions // Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, pp. 44-54. Minneapolis, Minnesota, 1994.
14. *Брой М.* Информатика, Часть 3. М.: Диалог-МИФИ, 1996.
15. *Смит Д.М., Смит Д.К.* Абстракции баз данных: агрегация и обобщение // СУБД, 1996. N 2. 141-160.
16. *Буч Г.* Объектно – ориентированный анализ и проектирование с примерами приложений на C++. М.: Издательство Бином, 1998.
17. *Fowler M., Scott K.* UML distilled applying the standard object modeling language. Reading: Addison Wesley, 1997.

Поступила в редакцию
05.05.2001