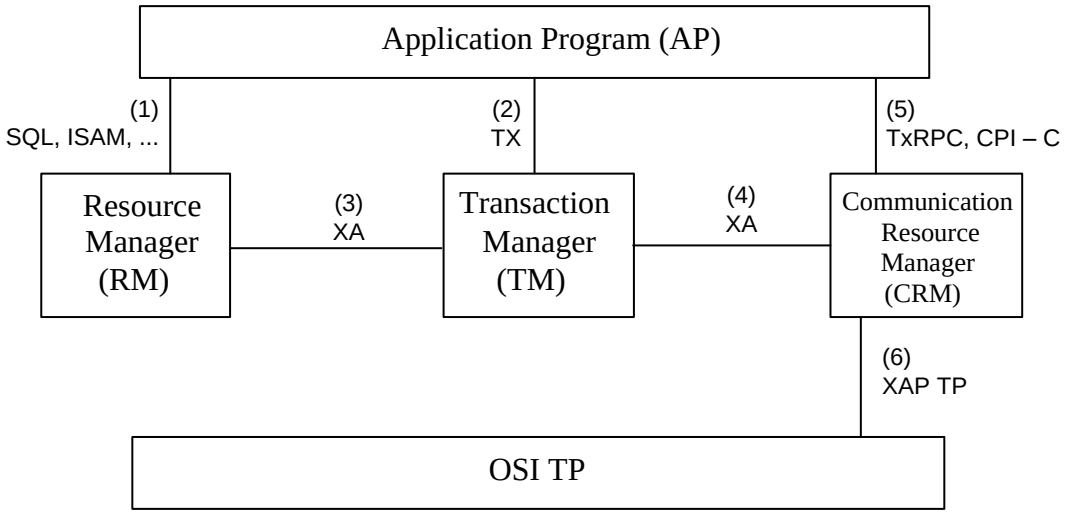
Модель обработки распределенных транзакций X/Open (X/Open DTP)

Эта модель используется как стандартная основа для любой реализации системы управления распределенными транзакциями. Она была разработана европейским консоциумом X/Open.

Модель DTP включает в себя несколько компонентов и определяет интерфейсы между ними (рис. 1).



AP – прикладная программа, которая участвует в распределенной транзакции. Она определяет моменты начала и окончания транзакции, взаимодействует с менеджерами ресурсов (RM), с координатором распределенных транзакций (TM) и с другими прикладными программами, вовлеченными в одну распределенную транзакцию, через компонент CRM.

TM – координатор распределенных транзакций, который обеспечивает атомарность распределенной транзакции путем синхронизации моментов начала и окончания распределенной (глобальной) транзакции с моментами начала и окончания локальных транзакций, поддерживаемых менеджерами ресурсов (RM). TM может взаимодействовать с удаленными координаторами через компонент CRM.

RM – это менеджер ресурсов, который обеспечивает транзакционный доступ к информации (реляционная СУБД, система обмена сообщениями, файловая система и пр.).

CRM – этот компонент обеспечивает возможность взаимодействия прикладных программ и координаторов с другими прикладными программами и координаторами по сети в рамках одной распределенной транзакции.

OSI TP – этот компонент обеспечивает низкоуровневую коммуникационную среду для организации сетевого взаимодействия между компонентами модели в рамках одной распределенной транзакции. Этот компонент входит в состав архитектуры ISO OSI и определяет сравнительно низкоуровневые сетевые протоколы, которые не являются частью модели X/Open, но на которые она опирается.

При взаимодействии компонентов X/Open DTP используются стандартизованные протоколы, также являющиеся частью модели:

(1) - протокол взаимодействия прикладной программы с менеджером ресурсов. Это может быть SQL, протокол ISAM (протокол последовательного доступа к файлам), стандартизованные консорциумом X/Open или что-то иное;

(2) - интерфейс взаимодействия прикладной программы с координатором, средствами которого прикладная программа может инициировать новую распределенную транзакцию, подтвердить или откатить ее, узнать состояние транзакции, изменить ее параметры и прочие. Использование этого протокола позволяет прикладной программе работать с глобальной транзакцией как с цельным объектом, в то время как на самом деле она состоит из множества локальных транзакций, поддерживаемых менеджерами ресурсов. В качестве стандартного протокола AP-TM используется TX;

(3) - протокол взаимодействия координатора транзакций с менеджером ресурсов. Используется координатором для управления началом и окончанием локальных транзакций, связанных с источниками данных, вовлеченными в глобальную транзакцию.

Этот протокол может быть использован и RM для динамической регистрации в распределенной транзакции. В качестве стандартного протокола TM-RM используется XA.

(4) – протокол взаимодействия координатора транзакции с компонентом CRM используется в случае, если одному координатору транзакции необходимо взаимодействовать с другими в рамках одной распределенной транзакции. В этом случае транзакция может охватывать, как говорят, несколько доменов. В качестве стандарта протокола TM-CRM является XA

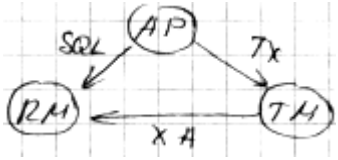
(5) - протокол взаимодействия прикладной программы с компонентом CRM используется для организации взаимодействия нескольких прикладных программ в рамках одной распределенной транзакции. Для взаимодействия AP-CRM предусмотрены стандартные протоколы TxRPC, CPI – C.

(6) – это протокол для сопряжения модели X/Open DTP с базовыми протоколами OSI TP. В качестве стандартного протокола используется XAP TP.

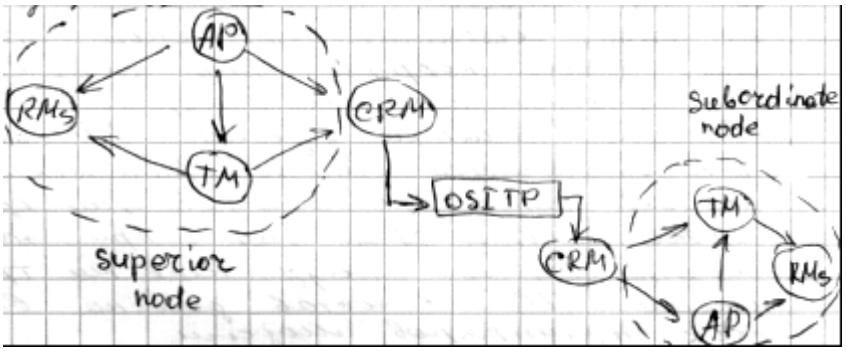
Экземпляр модели (instance of the model) – это набор, состоящий из одного приложения, одного координатора транзакций, одного или нескольких менеджеров ресурсов.

Транзакционный домен (TM domain) представляет собой множество приложений и источников данных, использующих один и тот же координатор транзакций, при этом координатор управляет сразу несколькими транзакциями. Таким образом, в состав домена входит несколько экземпляров модели.

В модели X/Open DTP контекст транзакции связан с потоком выполнения. Важность этого момента определяется тем, что с точки зрения отдельно взятого источника данных взаимодействие с ним ведется как со стороны координатора транзакций, так и со стороны прикладной программы (рис. 2). Источник данных определяет, что операции, выполняемые приложением, относятся к той или иной транзакции, на основе того, что они выполняются в том же потоке, в котором координатором транзакций была инициирована эта транзакция.



Экземпляры модели, связанные с разными координаторами транзакций, относятся к разным доменам. В общем случае возможно взаимодействие между координаторами транзакций и приложениями, относящимися к разным доменам. При этом между взаимодействующими экземплярами модели выстраиваются отношения ведущий (superior) - подчиненный (subordinate) и образуются структуры типа дерева (рис. 3).



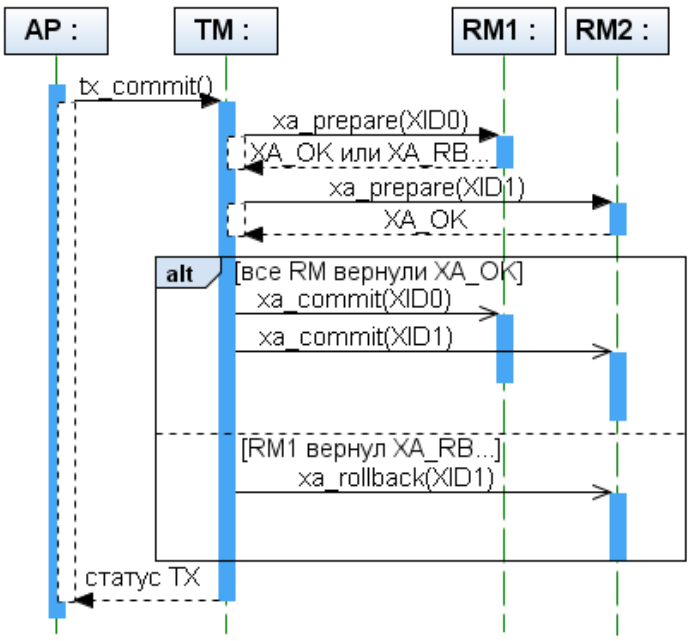
Один из узлов дерева при этом считается главным (root node) – это тот узел, в котором глобальная транзакция была инициирована. Завершить транзакцию может только тот узел, который ее инициировал, подчиненные узлы завершить ее не могут.

Ветвь распределенной транзакции (transaction branch) – это часть глобальной транзакции, которая относится к конкретному координатору и конкретному источнику данных, которые взаимодействуют в рамках этой глобальной транзакции.

Протокол двухфазной фиксации транзакций

При подтверждении транзакции используется двухфазный протокол (two-phase commit, 2PC), состоящий в том, что подтверждение осуществляется в 2 этапа (рис. 4):

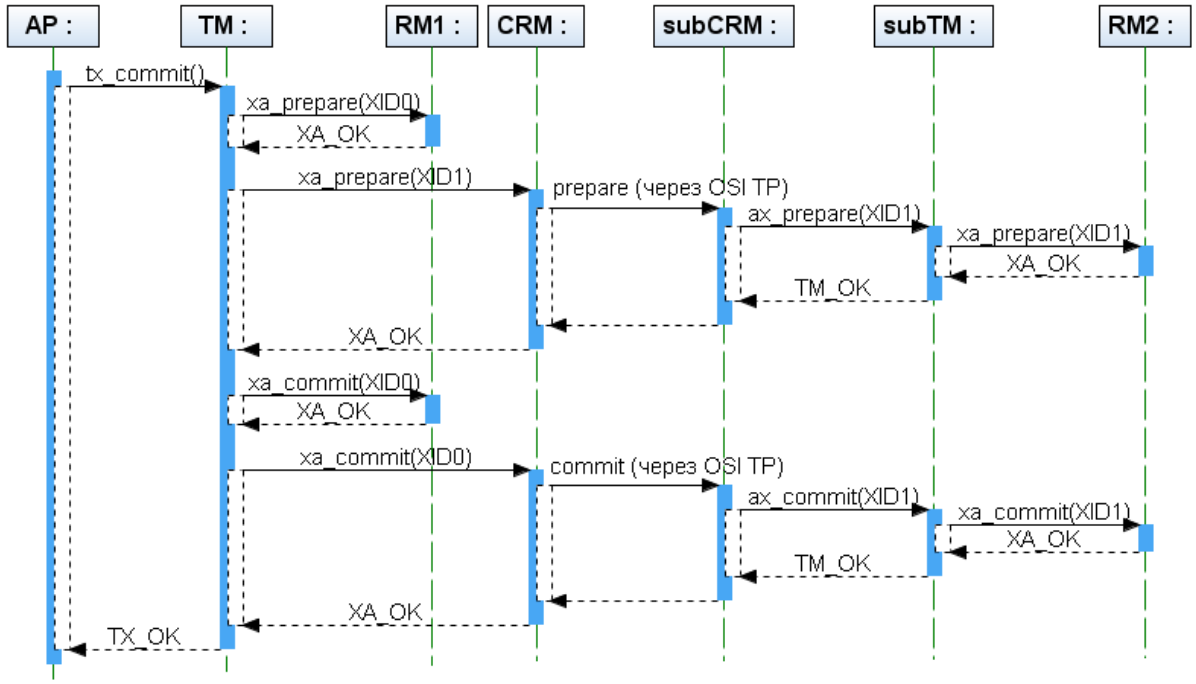
1. подготовка (prepare)
2. собственно подтверждение (commit)



На первом этапе координатор опрашивает все связанные с ним источники данных, вовлеченные в подтверждаемую распределенную транзакцию, готовы ли они характеризовать успешное подтверждение своей ветви. Если источник данных гарантирует успешное подтверждение, он отвечает на запрос положительно, иначе отрицательно.

При этом, если хотя бы один источник данных ответил отрицательно, то координатор выполняет не подтверждение, а откат транзакции, причем откат выполняется во всех источниках данных, вовлеченных в глобальную транзакцию. Если все источники данных

отвечали на запрос положительно, то выполняется вторая фаза – подтверждение. Общий результат завершения транзакции сообщается координатором приложению.



На рис. 5 отображена ситуация, когда распределенная транзакция пересекает границы одного транзакционного домена.

Предусмотрено две возможности для оптимизации двухфазного протокола:

1. если с конкретным источником данных в ходе глобальной транзакции выполнялись только операции чтения данных, то он может закончить свое участие в глобальной транзакции уже после первого этапа;
2. если в глобальной транзакции участвует только один источник данных, то вместо двухфазного протокола может использоваться обычный (однофазный) протокол подтверждения.

Модель X/Open DTP рассматривает три варианта отката глобальной транзакции:

1. явный откат (explicit rollback) – приложение явно инициирует откат;
2. неявный откат (implicit rollback) – приложение явно инициировало подтверждение транзакции, а в ходе первой фазы координатором был получен хотя бы один отрицательный ответ, что привело к откату, а не к подтверждению;
3. подразумеваемый откат (presumed rollback) – происходит при перезапуске источника данных после сбоя, произошедшего в нем в ходе глобальной транзакции.

Существенная черта 2PC – блокирующий характер протокола. Отказы могут происходить, в частности, на стадии подтверждения транзакции. Единственный способ выявления сбоев – это ожидание сообщения в течение определенного промежутка времени. Если время ожидания исчерпано, то ждущий процесс (координатор или источник данных) следует протоколу терминирования, который предписывает для такой ситуации способ обработки транзакции, находящейся в стадии завершения. Неблокирующий протокол фиксации – это такой протокол, терминирующая часть которого при любых обстоятельствах способна определить, что делать с транзакцией в случае сбоя. При использовании 2PC, если в период сбора голосов (подготовки) сбой происходит и на координирующем узле, и на одном из участников, оставшиеся узлы не способны решить между собой судьбу

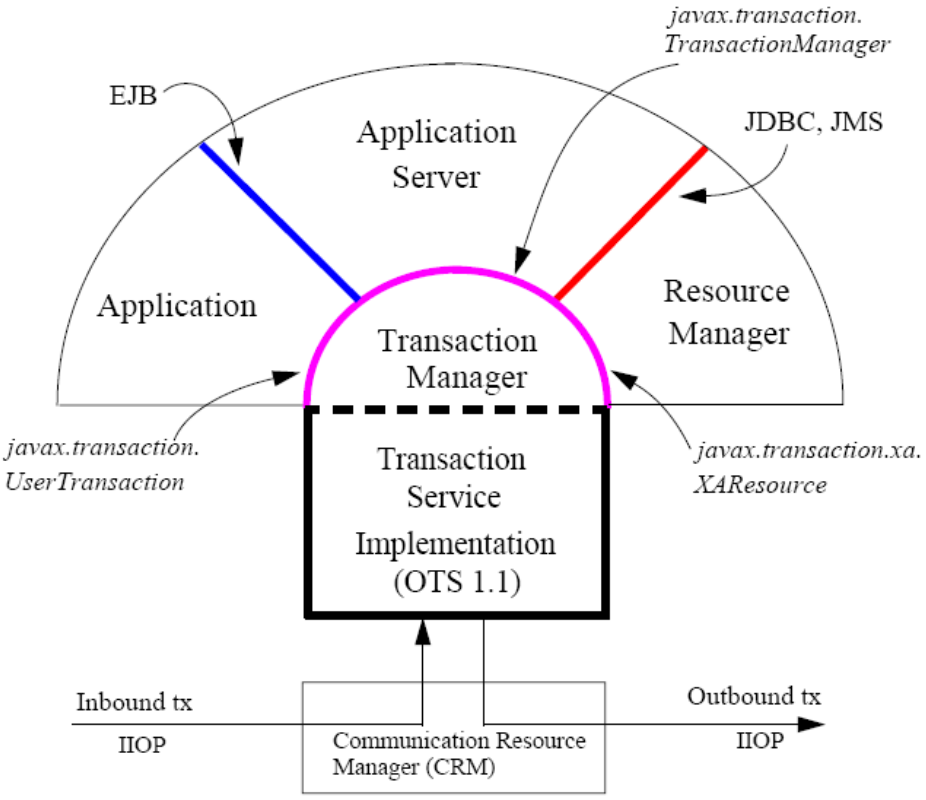
транзакции и вынуждены оставаться в заблокированном состоянии, пока не восстановится либо координатор, либо отказавший участник.

Модель X/Open DTP предусматривает возможность эвристического завершения транзакции (heuristic transaction completion) для частичного решения проблемы блокирования. В определенных случаях локальная транзакция в источнике данных может завершиться автоматически подтверждением или откатом до того, как будет получен соответствующий сигнал от координатора. Например, это может произойти, когда между первой и второй фазами подтверждения произошла большая временная задержка (по истечении некоторого таймаута). Говорят, что источник данных принял эвристическое решение (heuristic decision). Зачастую оно принимается без учета текущего состояния глобальной транзакции.

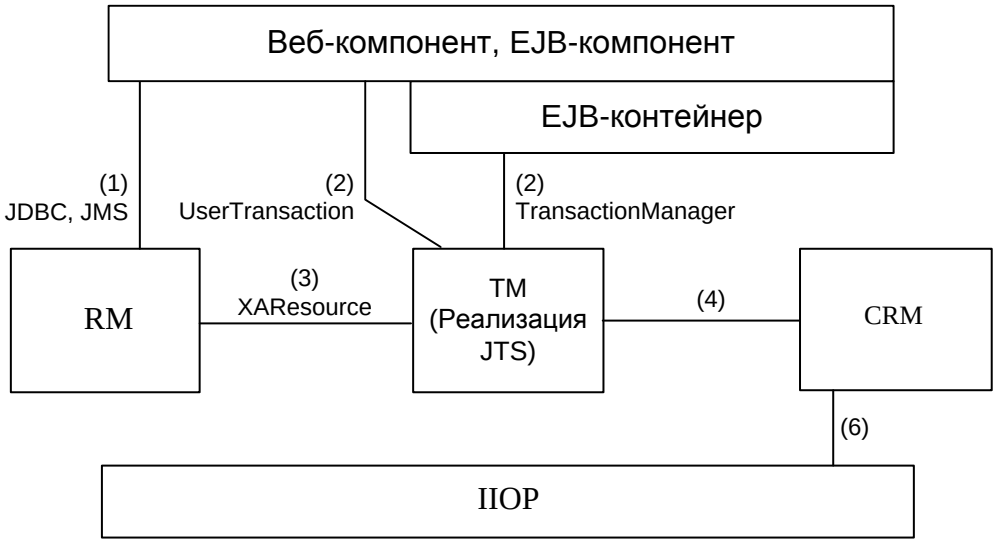
При этом возможно, что позднее будет инициирован откат или подтверждение глобальной транзакции, что может привести к несоответствию решения, принятого эвристическим источником данных, и решением, принятым приложением, инициировавшего завершения глобальной транзакции. В этом случае транзакционные свойства глобальной транзакции могут нарушаться. Координатор транзакций информирует приложение о том, были ли приняты эвристические решения или нет и, совпадали ли они с решением, принятым приложением или нет (последний вариант называется mixed mode). Возможен вариант, когда координатор транзакции не может определить, совпало ли решение, принятое источником данных эвристически с решением принятым приложением или нет.

Реализация модели X/Open DTP на платформе Java EE

На рис. 6 приведена архитектура системы обработки транзакций, построенной на платформе Java EE, взятая из спецификации на службу управления транзакциями JTS (Java Transaction Service).



Чтобы лучше представить себе, каким образом компоненты платформы Java EE отображаются на элементы модели X/Open DTP, приведем другую иллюстрацию (рис. 7).



Рассмотрим взаимодействие с СУБД в рамках распределенной транзакции с использованием интерфейса JDBC.

Для соединения с РСУБД используется объект типа `java.sql.Connection`, получаемый от менеджера драйверов (класс `java.sql.DriverManager`) или от источника данных (интерфейс `javax.sql.DataSource`) – первое характерно для независимых приложений, а второе – для приложений, выполняющихся в рамках сервера приложений. Однако через интерфейс `Connection` невозможно получить объект типа `javax.transaction.xa.XAResource`, обеспечивающий участие источника данных (СУБД) в глобальной транзакции.

Чтобы СУБД могла принимать участие в распределенной транзакции, для установки соединения с ней следует использовать особый тип источника данных – `javax.sql.XADataSource`. Работа с ним отличается от обычного источника данных типа `DataSource`. Так, для получения соединения (объекта типа `Connection`) используется следующая последовательность вызовов:

```
Connection conn = xaDataSource.getXAConnection().getConnection();
```

При работе в рамках сервера приложений компонент получает не тот объект источника данных (типа `DataSource`, `XADataSource`, либо `ConnectionPoolDataSource`), который был создан при конфигурировании сервера приложений, а обертку над ним, реализующую интерфейс `DataSource`.

Таким образом, с одной стороны, код компонента не зависит от типа выполняемой транзакции (локальная или глобальная), но с другой стороны, это может приводить к созданию источников данных неподходящих типов при установке приложения на сервер и, как следствие, к ошибкам в работе приложения.

Оптимизация локальных транзакций на платформе Java EE

Рассмотренная выше инфраструктура платформы Java EE для управления распределенными транзакциями ориентирована на общий случай, когда в рамках одной транзакции участвуют несколько разнородных менеджеров ресурсов. Тем не менее, в очень большом числе случаев в транзакцию вовлечен только 1 источник данных, причем он может и не поддерживать возможность работы в рамках распределенной транзакции (non-XA data source). Платформа позволяет использовать такие источники и работать с ними точно так же как с источниками, поддерживающими распределенные транзакции (XA data source), с тем лишь ограничением, что в рамках транзакции может быть задействован только один non-XA data source.

В терминах сервера приложений такие транзакции называются локальными. Транзакции, использующие хотя бы один XA data source, называются глобальными.

Несмотря на то, что приложение с локальной транзакцией работает точно так же, как с глобальной, сервер приложений при этом расходует гораздо меньше ресурсов, чем при обработке глобальной транзакции. В силу этого контейнер может оптимизировать свою работу в этом случае, а приложения могут работать быстрее – в этом и состоит оптимизация локальных транзакций (local transaction optimization).

При использовании глобальных транзакций, охватывающих только один источник данных (XA data source), сервер приложений может использовать оптимизацию, предусмотренную моделью X/Open DTP, когда при подтверждении глобальной транзакции используется не двухфазный протокол, а однофазный.