1、两阶段提交及JTA <https://www.cnblogs.com/jasongj/p/5727897.html>

（1）两阶段提交主要是保证了分布式事务的原子性。

（2）在分布式系统中，各个节点（或者事务参与方）之间在物理上相互独立，通过网络进行协调。每个独立的节点（或组件）由于存在事务机制，可以保证其数据操作的ACID特性。但是，各节点之间由于相互独立，无法确切地知道其经节点中的事务执行情况，所以多节点之间很难保证ACID，尤其是原子性。

如果要实现分布式系统的原子性，则须保证所有节点的数据写操作，要不全部都执行（生效），要么全部都不执行（生效）。但是，一个节点在执行本地事务的时候无法知道其它机器的本地事务的执行结果，所以它就不知道本次事务到底应该commit还是 roolback。常规的解决办法是引入一个“协调者”的组件来统一调度所有分布式节点的执行。

2、分布式系统，事务最终一致性解决方案

http://iamzhongyong.iteye.com/blog/2240891

3、Atomikos打印的日志不能够关闭，Atomikos通过这些日志来保障事务过程。atomikos是通过在应用的目录下生成日志文件来保证，如果失败，在重启后可以通过日志来完成未完成的事务。

4、普通JTA需要服务器进行支持，Atomikos不需要服务器进行支持。

5、pom中需要添加Atomikos的依赖包：

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-jta-atomikos</artifactId>

</dependency>

6、通过bean的方式注入：AtomikosDataSourceBean

7、创建UserTransactionManager、UserTransaction加入到JtaTransactionManager中。

UserTransactionManager userTransactionManager = new UserTransactionManager();

UserTransaction userTransaction = new UserTransactionImp();

JtaTransactionManager(userTransaction, userTransactionManager);

8、atomikos 是实现了XA的一种分布式事务处理工具。

9、Atomikos支持XA(全局事务)和NON-XA(非全局事务)。

10、想使用Spring带给我们的@Transactional便利，就需要配置一个JtaTransactionManager，而该JtaTransactionManager是需要一个userTransaction实例的。

11、具体就是需要一个xaDataSourceClassName，我们这里使用的是mysql支持xa的的MysqlXADataSource，它实现了javax.sql.XADataSource接口，即可以产生XAConnection连接。

12、XA接口提供资源管理器(TM)与事务管理器（RM）之间进行通信的标准接口。XA协议包括两套函数，以xa\_开头的及以ax\_开头的。

（1）xa\_open，xa\_close：建立和关闭与资源管理器的连接。

（2）xa\_start，xa\_end：开始和结束一个本地事务。

（3） xa\_prepare，xa\_commit，xa\_rollback：预提交、提交和回滚一个本地事务。

（4）xa\_recover：回滚一个已进行预提交的事务。

（5）ax\_开头的函数使资源管理器可以动态地在事务管理器中进行注册，并可以对XID(TRANSACTION IDS)进行操作。

（6）ax\_reg,ax\_unreg；允许一个资源管理器在一个TMS(TRANSACTION MANAGER SERVER)中动态注册或撤消注册。

13、Atomikos TransactionsEssentials是一个非常快速的嵌入式事务管理器，这就意味着，您不需要另外启动一个单独的事务管理器进程（不要查找任何的bin文件夹）。相反，您的应用服务器将有它自己的intra-VM事务管理器。

14、配置需求：至少Java1.5 jdk，并且最少128M的内存。

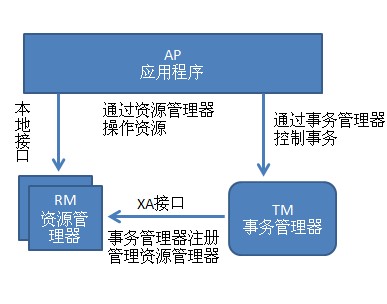
15、X/Open DTP 定义了三个组件： AP，TM，RM

AP(Application Program)：也就是应用程序，可以理解为使用DTP的程序

RM(Resource Manager)：资源管理器，这里可以理解为一个DBMS系统，或者消息服务器管理系统，应用程序通过资源管理器对资源进行控制。资源必须实现XA定义的接口

TM(Transaction Manager)：事务管理器，负责协调和管理事务，提供给AP应用程序编程接口以及管理资源管理器

其中，AP 可以和TM 以及 RM 通信，TM 和 RM 互相之间可以通信，DTP模型里面定义了XA接口，TM 和 RM 通过XA接口进行双向通信，例如:TM通知RM提交事务或者回滚事务，RM把提交结果通知给TM。



Atomikos中文说明：<https://blog.csdn.net/sun8288/article/details/8674016>

16、XA连接是一个JTA事务中的参与者。XA连接不支持JDBC的自动提交特性。也就是说应用程序不必在xadatasource[XA]连接上调用Java.sql.Connection.commit()或java.sql.Connection.rollback()；而应用程序应该使用UserTransaction.begin(),UserTransaction.commit()和UserTransaction.rollback()。