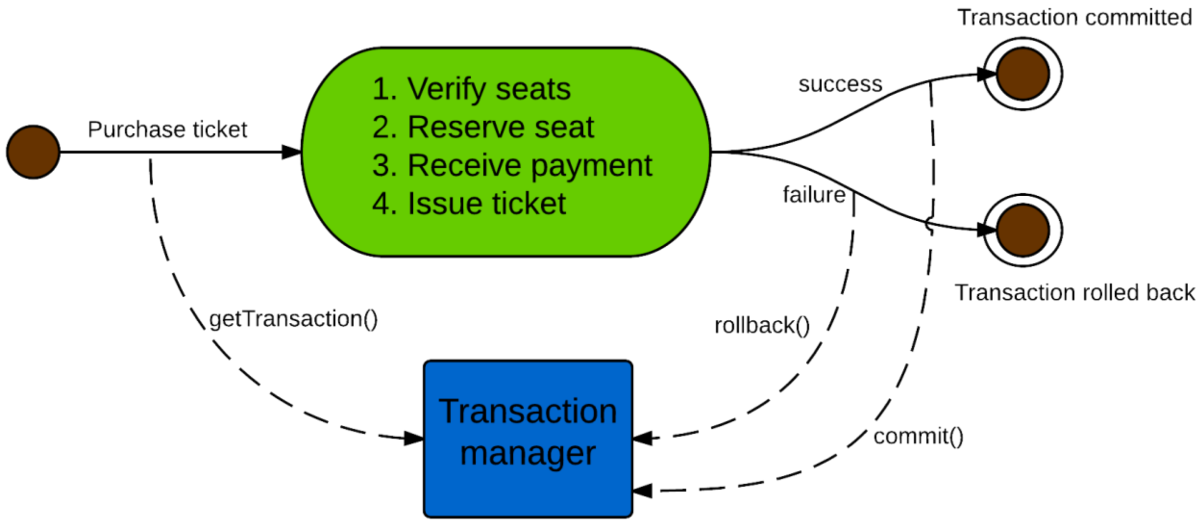
SPRING TRANSACTIONS

PROGRAMMATIC TRANSACTION MANAGEMENT

MARCH, 2015

Employing PlatformTransactionManager



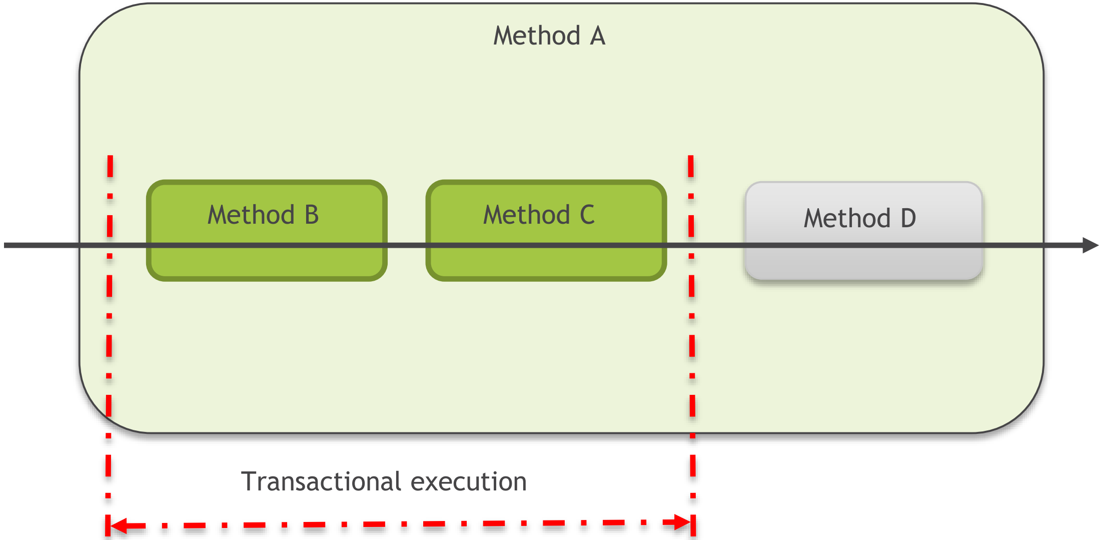
Sample programmatic transaction

```
DefaultTransactionDefinition def =
                                        new DefaultTransactionDefinition();
def.setName("SomeTxName");
def.setPropagationBehavior(TransactionDefinition.PROPAGATION_REQUIRES_NEW);
def.setIsolationLevel(TransactionDefinition.ISOLATION_REPEATABLE_READ);

TransactionStatus status = txManager.getTransaction(def);
try {
  // execute your business logic here
}
catch (MyException ex) {
  txManager.rollback(status);
  throw ex;
}
txManager.commit(status);
```

Catching transaction manager exceptions

```
DefaultTransactionDefinition def =
                                        new DefaultTransactionDefinition();
def.setName("SomeTxName");
def.setPropagationBehavior(TransactionDefinition.PROPAGATION_REQUIRES_NEW);
def.setIsolationLevel(TransactionDefinition.ISOLATION_REPEATABLE_READ);

TransactionStatus status = txManager.getTransaction(def);
try {
  // execute your business logic here
}
catch (TransactionTimedOutException ex) {
  //handle timeout properly
  txManager.rollback(status);
}
catch (MyException ex) {
  txManager.rollback(status);
  throw ex;
}
txManager.commit(status);
```

Custom transaction boundaries



Using TransactionTemplate

```
transactionTemplate.execute(new TransactionCallbackWithoutResult() {
 protected void doInTransactionWithoutResult(TransactionStatus status) {
  try {
   updateOperation1();
   updateOperation2();
  } catch (SomeBusinessExeption ex) {
   status.setRollbackOnly();
  }
 }
});
```

```
<bean id=“transactionTemplate”
               class=“org.springframework.transaction.support.TransactionTemplate”>
    <property name=“isolationLevelName” value=“ISOLATION_READ_UNCOMMITTED”/>
    <property name=“timeout” value=“30”/>
</bean>
```

Summary

Pros:

- Simple and concise API
- Fine grained control over transaction boundaries
- Flexible commit and rollback criteria
- Ability to provide fallback by handling TxManager exceptions

Cons:

- Error prone
- Decreased readability
- More boilerplate code
- Transaction management is mixed with business code

THANK YOU!

MAKSYM_GOVORISCHEV@EPAM.COM

MARCH, 2015