Sterling Commerce

Platform Transaction Management

Platform IFC 1.2

Sterling Commerce
An IBM Company

Using Transactional Processing

Overview

The Application supports extended and coordinated database transactions. In an extended transaction, there are two ways transactions can be configured to span across multiple steps in a business process. A coordinated transaction is one that is extended to include other global transaction resources. The Application supports transactions into certain adapters.

Note: Transactional processing is not available or supported for use with iSeries.

Configuring Transactional Processing

There are two ways to enable transactions in the Application so that when a rollback occurs, no data is committed; data returns to its pre-process state. You can either use the Begin and End Transaction Services in your business processes or you can enable transactions during the Business Process check-in.

Note: Within transactions, if a service is using data produced from a previous service (it may be internal to the Application) and the databasePool transaction was set to false (databasePool.transaction is defined in jdbc.properties), sometimes the Application appears locked. This lock situation is caused by the transaction isolation level setting in the database. For example, read_committed (default for DB2 and MYSQL), because the transaction of the business process has not been committed, the database can not operate on the data outside the scope of the current transaction. To avoid this lock situation, use the Begin and End Transaction service in the business process and have the service that does not use transactional pool be excluded from the transaction.

Using the Business Process Check-in Transaction Option

During the business process check-in, you are given an opportunity to mark the business process for transaction processing. Use this option, Enable Transaction, when you need to generate the business process output into a single transaction. For example, if all of the steps/services configured in the business process need to respect transactional boundaries, then the data related to the process exists as a whole or nothing in the database. In addition, this enables a clean resume/restart of a business process without any errors and maintains transaction integrity of data spanning all the steps in the business process.

Configuring On Fault Processing for Transactions

During the business process check-in, you are given the opportunity to have the business process commit all work to the database, at the time an error is generated. This, Commit All Steps when there is an error, option is only valid when the BPML is configured with an onfault block and the Enable Transaction is selected. With this option you can either rollback or commit all work prior to the error before the onfault block is called.

If this option is selected, all steps prior to the error are committed to data base and the business process keeps running on the onfault block path.

If this option is not selected, all steps prior to the error will be rolled back and the onfault block path is called.

Using the Begin and End Transaction Services

The Application provides you with a way to form a transactional block within the business process. The Begin Transaction and End Transaction services:

- Provide the ability to group multiple sequential steps into a single transaction
- Provide the ability to commit or roll back all of the steps at the same time
- Reduce the labor-intensive and problematic manual recovery costs and measures.

The following constraints apply to the use of extended transactions:

- When a transaction is entered, the business process is locked to its current node. That is, it will continue executing on the same engine thread for the duration of the transaction.
- The business process cannot branch after it is engaged in an extended transaction.

Services/Adapters That Support Transactional Processing

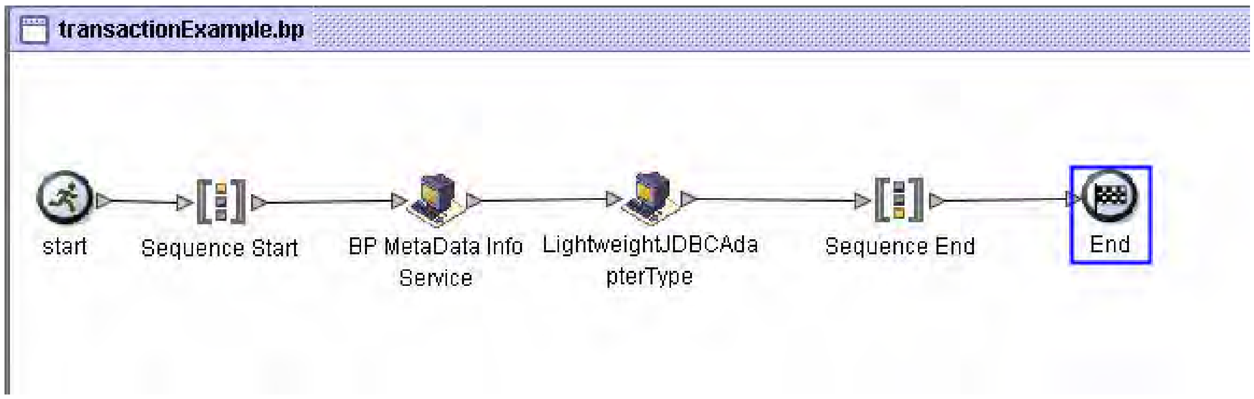
The following services and adapters can complete extended and coordinated database transactions:

- JDBC Adapter
- Lightweight JDBC Adapter
- Lock Service
- Mailbox Add Service
- Mailbox Correlate Documents Service
- Mailbox Delete Mailbox Service
- Mailbox Delete Service
- Mailbox Evaluate All Automatic Routing Rules Service
- Mailbox Evaluate Routing Rule Service
- Mailbox Extract Abort Service
- Mailbox Extract Begin Service
- Mailbox Extract Commit Service
- Mailbox Query Service
- Mailbox List Service
- Mailbox Scheduled Delete Service
- Mailbox Update Service
- Release Service

- Dynamic Service Invoker

Example of Business Process that Supports Transactions

The following is an example of a business process that upon checkin, could be marked for enable transaction.



Code Example - Business Process that Supports Transactions

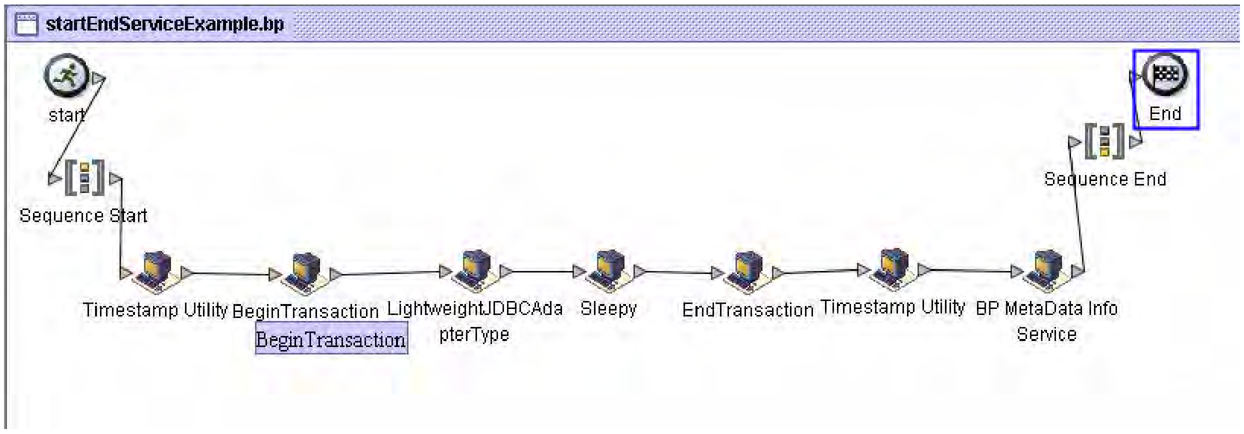
```
<process name="example_trans">
  <sequence>
    <operation name="BP MetaData Info Service">
      <participant name="BPMetaDataInfoService"/>
      <output message="Xout">
        <assign to="DISPOSITION">true</assign>
        <assign to="LINKAGE">true</assign>
        <assign to="CORRELATION">true</assign>
        <assign to="TRACE">true</assign>
        <assign to="." from="*"></assign>
      </output>
      <input message="Xin">
        <assign to="." from="*"></assign>
      </input>
    </operation>

    <operation name="LightweightJDBCAdapterType">
      <participant name="LightweightJDBCAdapterQuery"/>
      <output message="LightweightJDBCAdapterTypeInputMessage">
        <assign to="sql">INSERT INTO WORKFLOW_DATA (WF_INSTANCE_ID,DATA_KEY,DATA
_NAME) VALUES (?, ?,?) </assign>
        <assign to="param1" from="&apos;-1&apos;"></assign>
        <assign to="paramtype1">Integer</assign>
        <assign to="param2" from="&apos;TestRollBack_7859&apos;"></assign>
        <assign to="paramtype2">String</assign>
        <assign to="param3" from="&apos;RollbackTest&apos;"></assign>
        <assign to="paramtype3">String</assign>
        <assign to="pool">mysql</assign>
        <assign to="row_name">row</assign>
        <assign to="query_type">ACTION</assign>
        <assign to="result_name">result</assign>
        <assign to="." from="*"></assign>
      </output>
      <input message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>

  </sequence>
</process>
```

Example of Business Process that Includes Begin/End Service

The following is an example of a business process that includes the Begin Transaction and End Transaction service.



Code Example - Business Process that Includes Begin/End Service

```
<process name="mStepJDBCService">
<sequence>
<operation name="BP MetaData Info Service">
<participant name="BPMetaDataInfoService"/>
<output message="Xout">
<assign to="DISPOSITION">true</assign>
<assign to="LINKAGE">true</assign>
<assign to="CORRELATION">true</assign>
<assign to="TRACE">true</assign>
<assign to="." from="*"></assign>
</output>
<input message="Xin">
<assign to="." from="*"></assign>
</input>
</operation>

<operation name="LightweightJDBCAdapterType">
<participant name="LightweightJDBCAdapterQuery"/>
<output message="LightweightJDBCAdapterTypeInputMessage">
<assign to="sql">INSERT INTO WORKFLOW_DATA (WF_INSTANCE_ID,DATA_KEY,DATA
_NAME) VALUES (?, ?,?) </assign>
<assign to="param1" from="'-1'"></assign>
<assign to="paramtype1">Integer</assign>
<assign to="param2" from="'TestRollBack_7859'"></assign>
<assign to="paramtype2">String</assign>
<assign to="param3" from="'RollbackTest'"></assign>
<assign to="paramtype3">String</assign>
<assign to="pool">&DB_POOL;</assign>
<assign to="row_name">row</assign>
<assign to="query_type">ACTION</assign>
<assign to="result_name">result</assign>
<assign to="." from="*"></assign>
</output>
<input message="inmsg">
<assign to="." from="*"></assign>
</input>
</operation>
</sequence>
</process>
here is the on with start|endtransaction service, service between star|endtransaction
will share one transaction, i.e. if either one of lightweighjdbcservice, or
sleepservice failed , none of then will be persisted. the rest of service will be in
its own transaction.
<process name="simpleCommitOnErrorTest_01">
<sequence>
<operation name="Timestamp Utility">
<participant name="TimestampUtilService"/>
<output message="TimestampUtilServiceTypeInputMessage">
<assign to="action">current_time</assign>
<assign to="." from="*"></assign>
</output>
<input message="inmsg">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="BeginTransaction">
<participant name="BeginTransactionService"/>
<output message="Xout">
<assign to="START_TRANSACTION">TRUE</assign>
</output>
<input message="Xin">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="LightweightJDBCAdapterType">
<participant name="LightweightJDBCAdapterQuery"/>
<output message="LightweightJDBCAdapterTypeInputMessage">
<assign to="sql">INSERT INTO WORKFLOW_DATA (WF_INSTANCE_ID,DATA_KEY,DATA
_NAME) VALUES (?, ?,?) </assign>
<assign to="param1" from="'-1'"></assign>
<assign to="paramtype1">Integer</assign>
<assign to="param2" from="'TestRollBack_7859'"></assign>
<assign to="paramtype2">String</assign>
<assign to="param3" from="'RollbackTest'"></assign>
<assign to="paramtype3">String</assign>
<assign to="pool">&DB_POOL;</assign>
```

```
<assign to="row_name">row</assign>
<assign to="query_type">ACTION</assign>
<assign to="result_name">result</assign>
<assign to="." from="*"></assign>
</output>
<input message="inmsg">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="Sleepy">
<participant name="EngineTestSleep"/>
<output message="Xout">
<assign to="SLEEP_INTERVAL">20</assign>
<assign to="." from="*"></assign>
</output>
<input message="Xin">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="EndTransaction">
<participant name="EndTransactionService"/>
<output message="Xout">
<assign to="END_TRANSACTION">TRUE</assign>
</output>
<input message="Xin">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="Timestamp Utility">
<participant name="TimestampUtilService"/>
<output message="TimestampUtilServiceTypeInputMessage">
<assign to="action">current_time</assign>
<assign to="." from="*"></assign>
</output>
<input message="inmsg">
<assign to="." from="*"></assign>
</input>
</operation>
<operation name="BP MetaData Info Service">
<participant name="BPMetaDataInfoService"/>
<output message="Xout">
<assign to="DISPOSITION">true</assign>
<assign to="LINKAGE">true</assign>
<assign to="CORRELATION">true</assign>
<assign to="TRACE">true</assign>
<assign to="." from="*"></assign>
</output>
<input message="Xin">
<assign to="." from="*"></assign>
</input>
</operation>
</sequence>
</process>
```