

Overview

Overview of the BTM JTA Transaction Manager

The Bitronix Transaction Manager (BTM) is a simple but complete implementation of the JTA 1.0.1B API. The goal is to provide a fully working XA transaction manager that provides all services required by the JTA API while trying to keep the code as simple as possible for easier understanding of the XA semantics. This is BTM's strongest point compared to its competitors: when something goes wrong it is much easier to figure out what to do thanks to the great care placed in useful error reporting and logging.

Contents

- [Pragmatic description](#)
- [Current status](#)
 - [JDBC](#)
 - [JMS](#)
 - [JCA](#)

Pragmatic description

BTM is a perfect choice for a project using the Spring framework and willing to leverage [Spring's transactions capabilities](#) by using the [JtaTransactionManager](#) facade. You can safely mix JDBC and JMS accesses in a single unified transaction while you can use Spring's JDBC or JMS Templates with very good performance as BTM ships with efficient pools for both resource types.

It is also possible to integrate BTM in web containers like Tomcat or Jetty and get raw access to a JTA implementation to integrate for instance with Hibernate to get automatic session context management.

BTM has proved to be stable and mature enough to be used in production by at least two major corporations where it has been running flawlessly (and is still running) for months.

Current status

Currently BTM is very stable and usable. JDBC and JMS resources are working pretty well and recovery after crash works plain fine thanks to the embedded disk journal. It has been so since the early alpha releases because thorough testing is done before any release.

Normally, only XA aware resources can participate in global transactions (with a special exception for JDBC) which means there are three resource types that can be used:

JDBC

For JDBC, the driver must provide an implementation of the [javax.sql.XADataSource](#) interface. Please note that the underlying database must also provide some support to the driver one way or another. It is not possible to write a JDBC driver supporting XA for a database that does not support it. On the other hand, it is possible to write a driver that fakes XA support and this solution has been used by some vendors.

BTM also provides a way to allow any JDBC driver to take part in a XA transaction by emulating XA with the Last Resource Commit optimization. You can read more about this feature [in the documentation](#).

A list of databases that have been tested with BTM is being maintained [here](#).

JMS

For JMS, your vendor must provide an implementation of the [javax.jms.XAConnectionFactory](#) interface. The same remark as for JDBC holds: only implementations truly supporting XA (or trying to) have been considered.

A list of JMS servers that have been tested with BTM is being maintained [here](#).

JCA

JCA connectors could potentially be used too but BTM currently lacks support for them.