JMS Sampler

Here is a description of the components planed for the JMS sampler.

- JMS publish sampler
- JMS subscribe sampler
- Sample aggregator
- Graph to display publish, subscribe and total in terms of kb/sec
- generic client class used by both Publish and subscribe sampler

The two samplers will have common properties, which users must provide.

- URL - Context.PROVIDER_URL
- Initial context - Context.INITIAL_CONTEXT_FACTORY
- ConnectionFactory - server specific setting
- Topic - the specific topic for the test
- Iterations - number of iterations to aggregate
- use 1 connection for N producers/consumers (suggested by james)
- use N/2 connections for N producers/consumers (suggested by james)
- use N connections for N producers/consumers (suggested by james)
- Message type - TextMessage, ObjectMessage
- random message - similar to how webservice sampler can randomly select a message
- Text area - for message
- Use as monitor - this is similar to HTTPSampler for Tomcat5. when it is checked, the sample is set as monitor result, which is then used by the monitor to display the health and performance

The sample Aggregator is needed to aggregate several samples together, since a single publish will probably less than 1 millisecond, it's a good idea to run X iterations and record the time.

By having a publish and subscribe sampler, it means a single test plan can have two thread groups running concurrently. One thread group can have X threads publishing, while the thread group for the subscribers can have a different number of threads. this should allow for a wide variety of test scenarios. Using the same connection for N producers/consumers will measure the effect of connection on performance.

Here's some details about what parameters the sampler GUI will have.

1. radio buttuns: JNDI, manual

Parameters for JNDI

---

1. InitialContextFactory 2. provider url 3. connection factory 4. topic 5. user 6. password

Parameters for manual connections

---

1. concrete connection factory class 2. topic 3. user 4. password

Common parameters

---

1. message type: text or object 2. textarea 3. directory of files to choose randomly 4. specific file 5. use as monitor

After some discussion with James, it seems to make sense to treat the Session as the client. This way, a single connection may have multiple Sessions. I still need to work out the details of how measure the time for subscribers, since they will get notified of when new messages come in. For the first implementation, I'm going to by pass a connection factory. Once it the details of how subscribers will work, I'll move on to the connection factory.

For the connection factory, the plan right now is to make a real JMS connection pool. Most likely I'll borrow heavily from JDBC pool connection API and modify it to meet the needs of the JMS Sampler. This way, I can have it implement TestListener to setup and tear down the connection pool propertly. It would be bad to create a pool of connections to JMS at the start of a test, but not close those connections when the test ends.

Here are screen captures of the GUI so far.

---

http://www.apache.org/~woolfel/subscriber.png
http://www.apache.org/~woolfel/publisher.png