**SJMS-100-ClientJSE:**

**Codice da spiegare:**

JndiUtils: (da configurazione a classi Java per Producer e Consumer)  
Producer: (bean spedito su destinazione)  
Consumer: (modalita consumo)  
Eccezioni: (validazione consumer => per messaggio poison)

**PuntiControllo Demo:**

consumer.properties (demo queue/topic, trans e ack)  
producer.properties (demo queue/topic, trans e ack)  
consumerAction  
producerAction

**Demo protocollo (TC)**

**Modelli comunicazione (Test sia per Queue che per Topic )**- TC1: Durable (interrompere consumer e riprendere)  
- TC2:Consumo esclusivo(avviare piu thread consumer)  
- TC3: Consumo Polling (main/test classi)  
- TC4: Consumo ED (main/test classi)  
 **Test Acks singolo messaggio:**Verifica su Db quando viene cancellato messaggioCambiare auto o clientda consumer.properties  
Localizzare e spiegare message.acknowledge()  
Ack su Producer (send blocca fino a che il MOM non lo salva su DB, non testabile)- TC5: Polling props ack (auto e client) (test in debug e verifica in DB)  
- TC6: ED props Ack (auto e client) (test in debug e verifica in DB)

**Transazioni**Verifica su DB movimento chunk in conseguenza di commit, rollback e RE sia per Producer che per Consumer.  
Modificare impostazioni transazione su consumer.properties  
  
N.B  
Verificare da DB e UI (priorita UI)  
(a volte msg non in DB sono comunque nella retry cache della Queue)

TC7 Prod:   
Debug su .send e .commit.  
Verifica su DB.  
  
TC8 Prod: no Commit  
session.rollBack   
RE (fine thread producer perche’ non gestita)  
Verifica DB vuoto.  
  
- TC9: Consumer Polling RE Transazioni true/false (DEBUG)  
Trans =true,   
Verifica su Db che l ack viene dato per chunk e non per singolo.  
(anche se ack e’ AUTO non viene dato al momento del .receive() )  
In caso di RE o session.rollBack il chunk rimane in DB.

- TC10: ED e RE (conf DeadLetterQueue)  
Trans =true, ( indifferente in ML non ha accesso alla session jms)  
onMessage -> Eccezione (redelivery)  
Verifica da UI in ActiveMQ.DLQ dopo 20 tentativi (sembra)  
Dettagli:  
<http://activemq.apache.org/message-redelivery-and-dlq-handling.html>

**SPIEGAZIONI (EV SLIDES):**

**Forme di Ack: (da verificare in debug, quando viene rimosso dalla queue in DB):  
  
Ack consumer:**L’ack e’ un messaggio inviato dal consumer al MOM.  
Solo alla ricezione del ack il MOM e’ autorizzato a cancellare dal DB il corrispondente messaggio.  
Se il messaggio non risulta ricevuto (unacked) il MOM effettuera’ altri tentativi.   
(visibili nel JMSHeader redelivered=true)

**AUTO\_ACK**Il messaggio si considera ricevuto (e quindi cancellabile dal MOM) alla conclusione di  
1) return onMessage() del Listener  
2) receive() del Poller   
  
Non Ack se:  
1) onMessage va in eccezione  
2) errore delivery del MOM sul metodo receive (non simulabile)

**DUPS\_OK\_REDELIVERED:**L’ autoack aggiunge un overhead dovuto alla verifica che la delivery non venga provata piu di  
una volta.  
Questa opzione di ack si basa sul fatto che verranno fatti N tentativi di consegnare lo stesso messaggio, e ci si aspetta che il consumer scarti quelli che ha gia ricevuto.  
(gestione doppioni da codice, in cambio di migliori performances)

**CLIENT\_ACKNOWLEDGE:**L’ack viene dato esplicitamente da codice.

API: message.acknowledge()

ED: si puo’ indicare un punto precedente al default (uscita onMessage)  
PL: si puo’ indicare un punto successivo al default (receive restituisce)