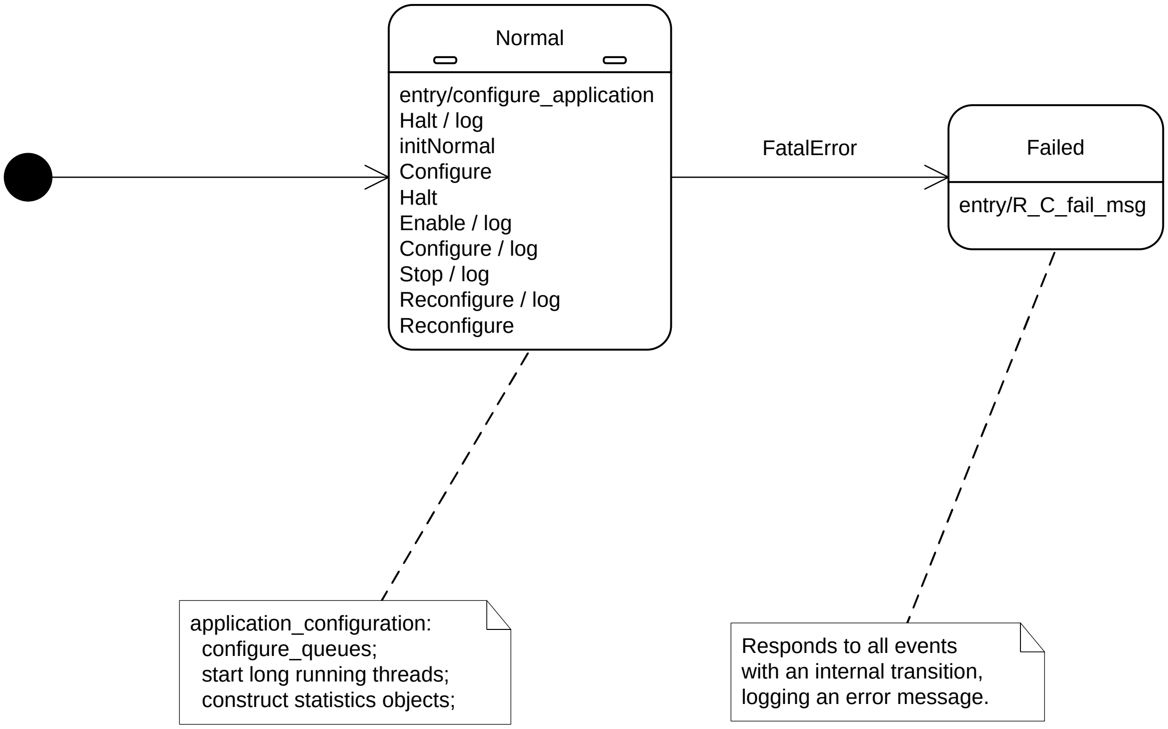
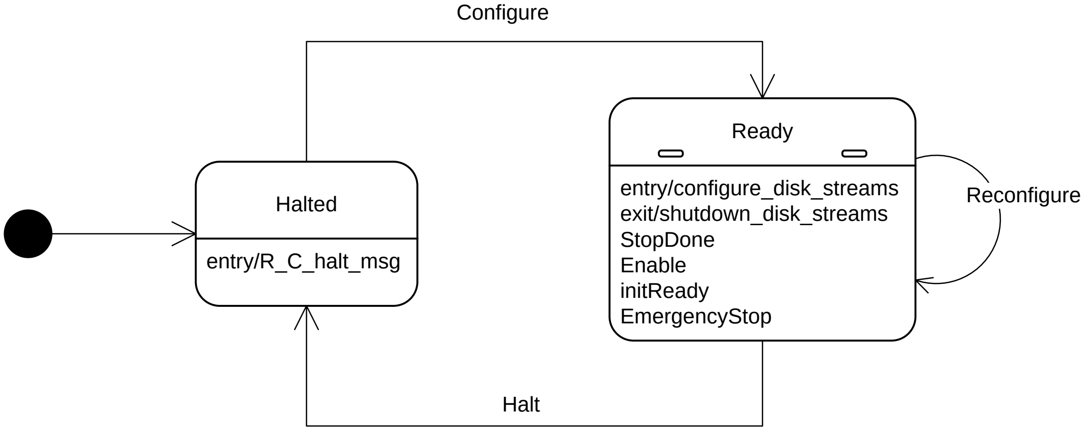
Top-level states



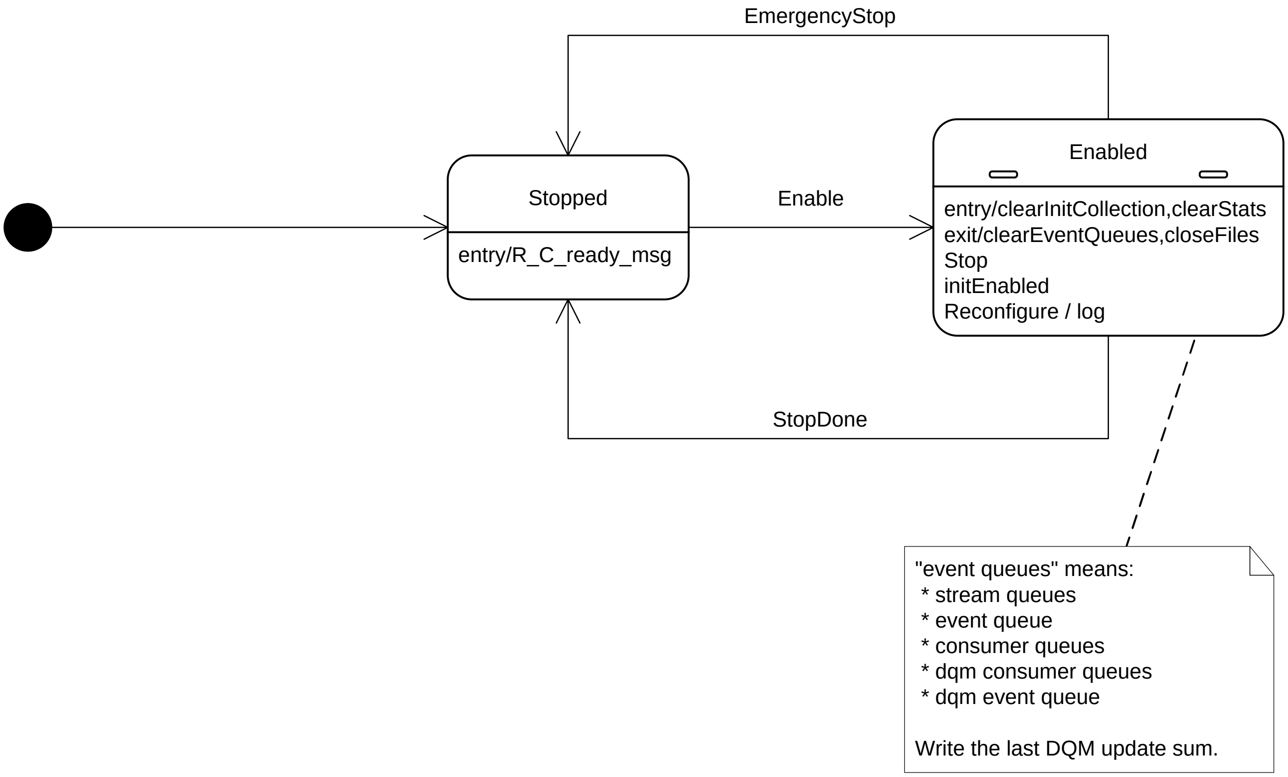
application_configuration:
  configure_queues;
  start long running threads;
  construct statistics objects;

Responds to all events
with an internal transition,
logging an error message.

Substates for *Normal* state.



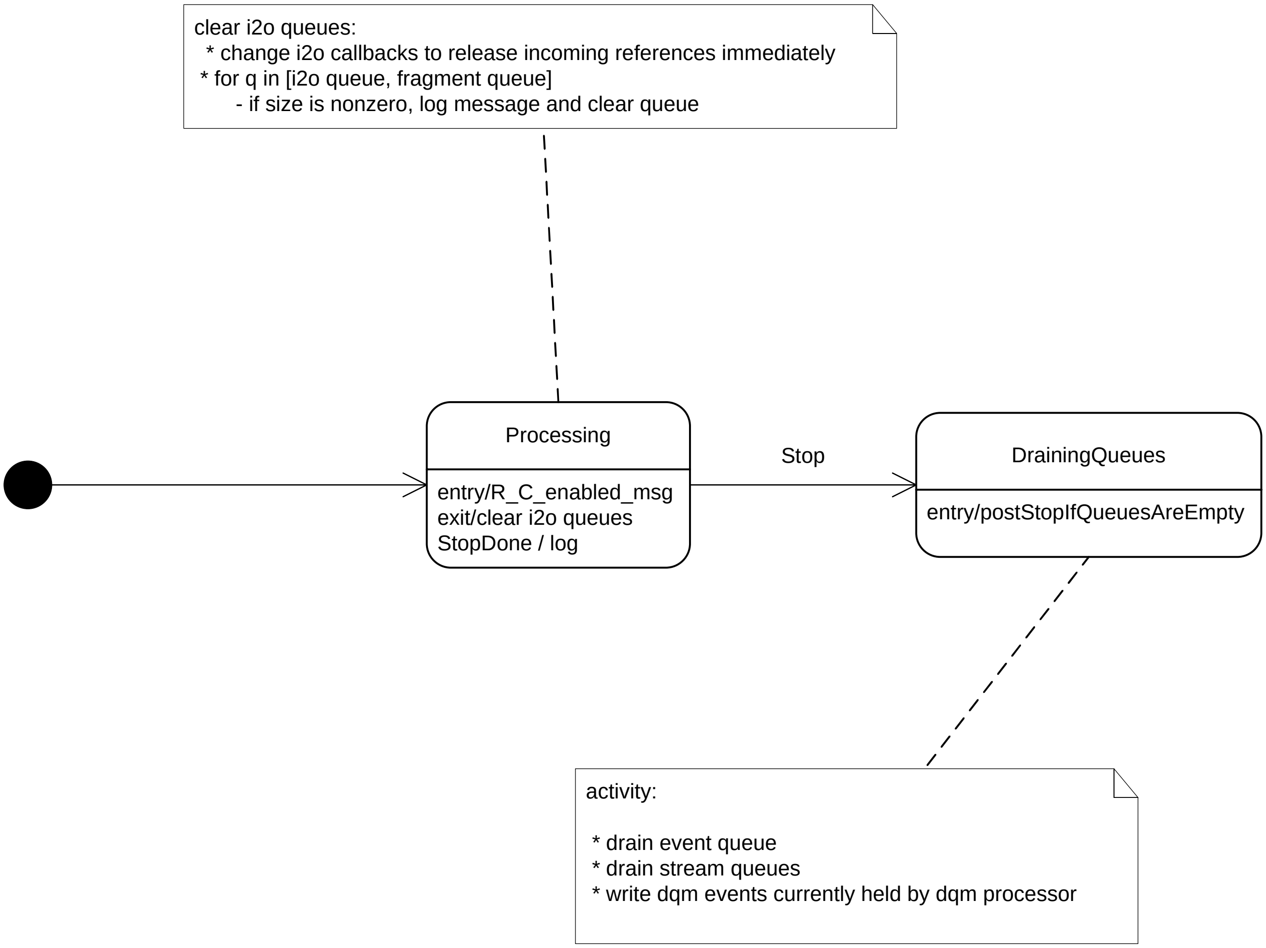
Substates for *Ready* state

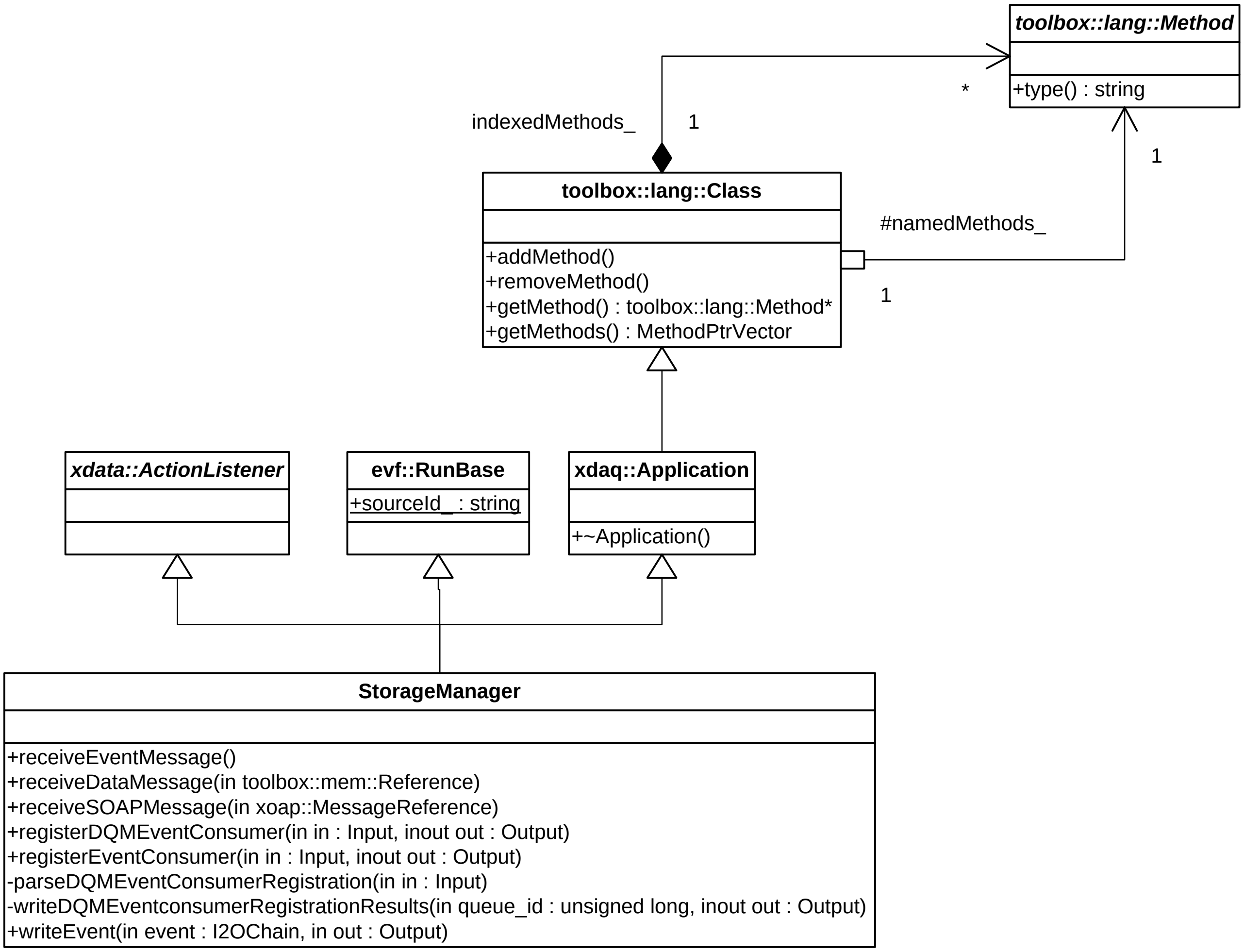


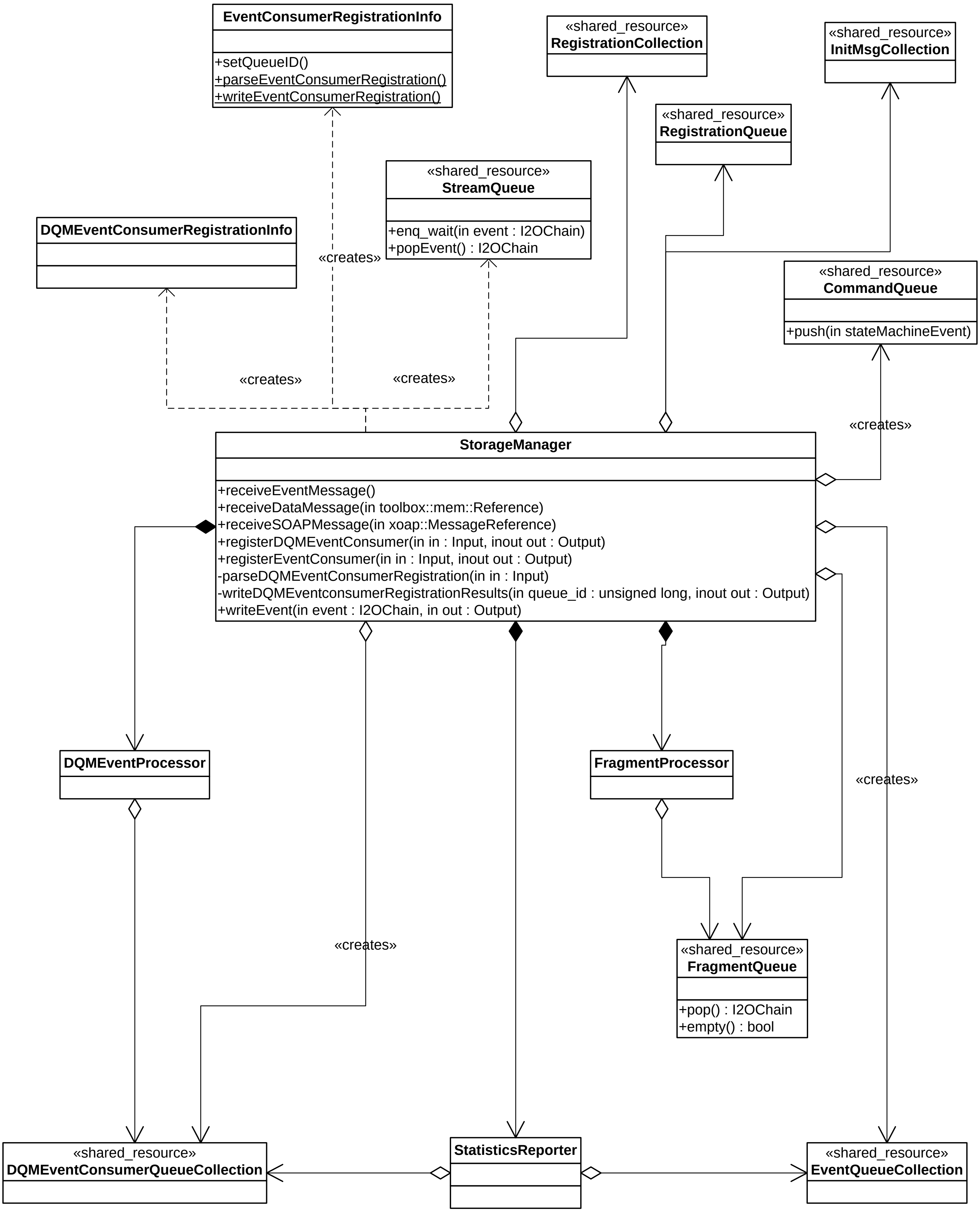
"event queues" means:
- stream queues
- event queue
- consumer queues
- dqm consumer queues
- dqm event queue

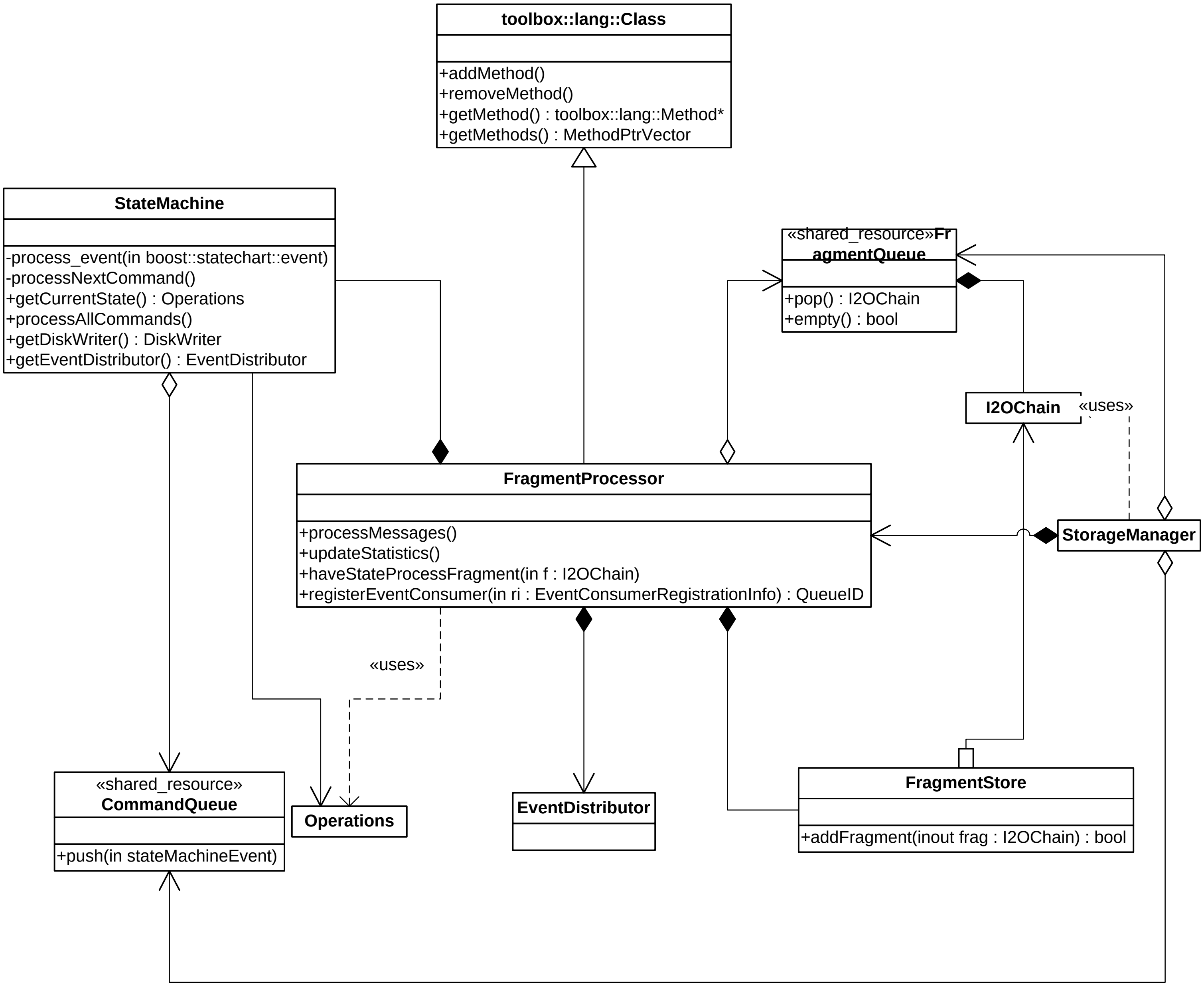
Write the last DQM update sum.

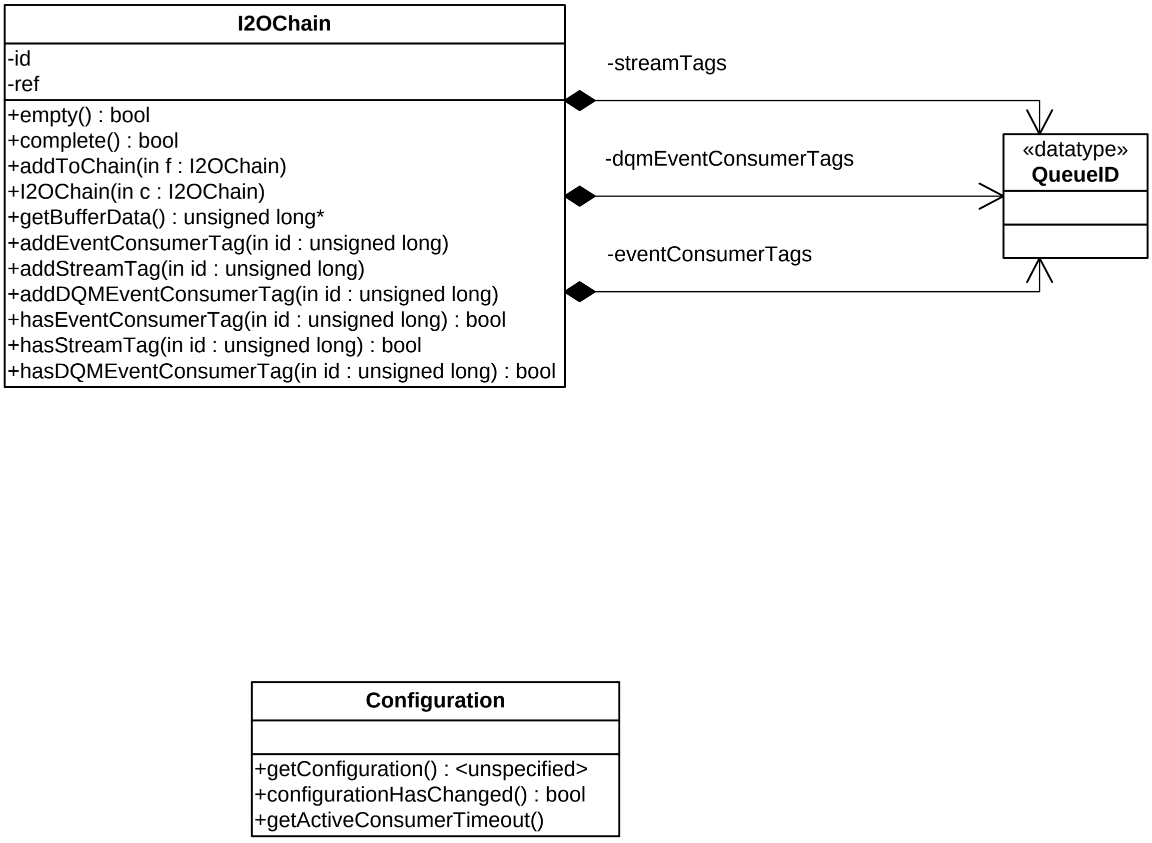
Substates for *Enabled* State

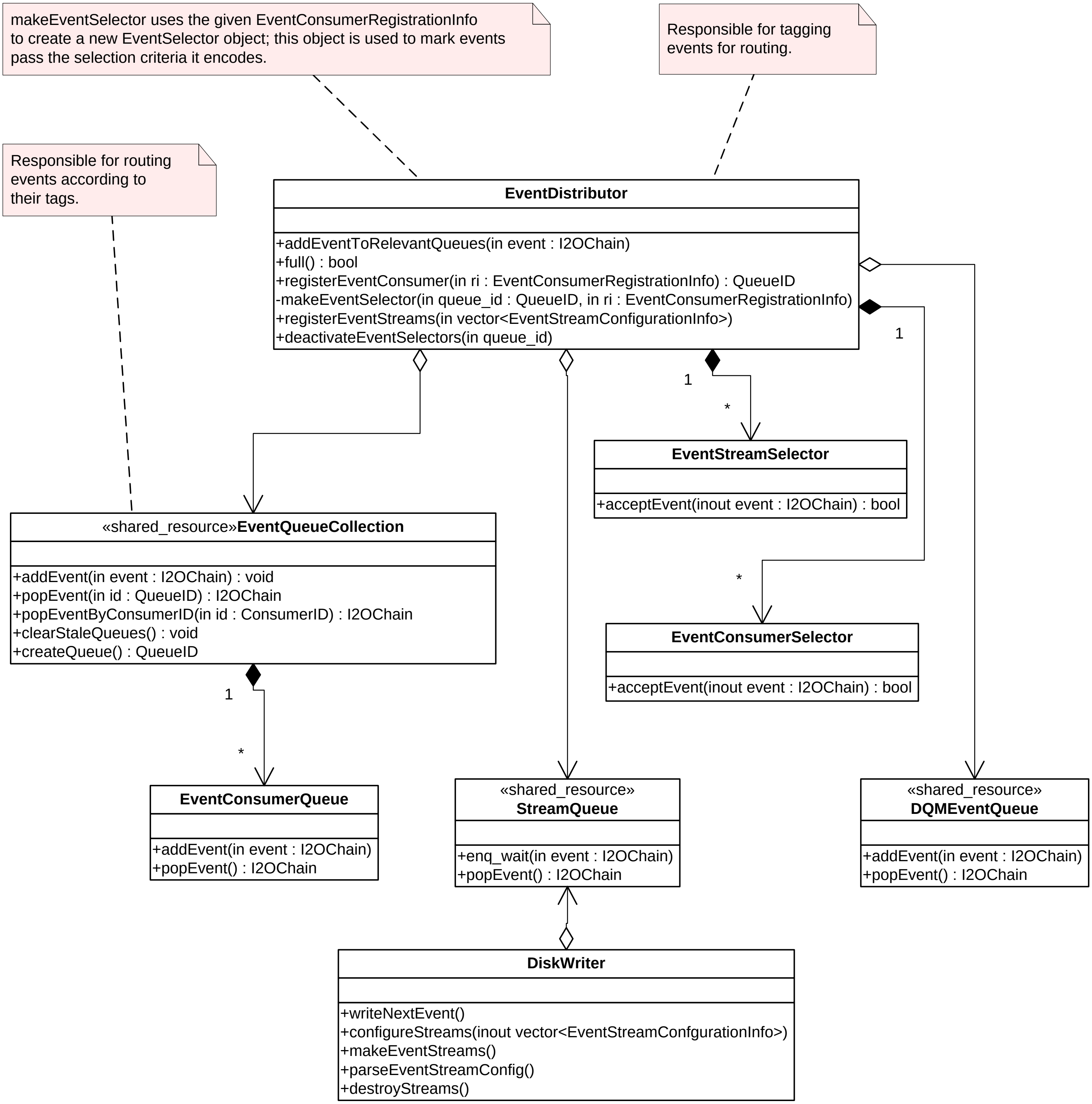

clear i2o queues:
* change i2o callbacks to release incoming references immediately
* for q in [i2o queue, fragment queue]
- if size is nonzero, log message and clear queue
Processing
entry/R_C_enabled_msg
exit/clear i2o queues
StopDone / log
Stop
DrainingQueues
entry/postStopIfQueuesAreEmpty
activity:
* drain event queue
* drain stream queues
* write dqm events currently held by dqm processor


toolbox::lang::Method
+type() : string
*
indexedMethods_
1
1
toolbox::lang::Class
+addMethod()
+removeMethod()
+getMethod() : toolbox::lang::Method*
+getMethods() : MethodPtrVector
#namedMethods_
1
xdata::ActionListener
evf::RunBase
+sourceId_ : string
xdaq::Application
+~Application()
StorageManager
+receiveEventMessage()
+receiveDataMessage(in toolbox::mem::Reference)
+receiveSOAPMessage(in xoap::MessageReference)
+registerDQMEventConsumer(in in : Input, inout out : Output)
+registerEventConsumer(in in : Input, inout out : Output)
-parseDQMEventConsumerRegistration(in in : Input)
-writeDQMEventconsumerRegistrationResults(in queue_id : unsigned long, inout out : Output)
+writeEvent(in event : I2OChain, in out : Output)


EventConsumerRegistrationInfo
+setQueueID()
+parseEventConsumerRegistration()
+writeEventConsumerRegistration()
«shared_resource»
RegistrationCollection
«shared_resource»
InitMsgCollection
«shared_resource»
RegistrationQueue
«shared_resource»
StreamQueue
+enq_wait(in event : I2OChain)
+popEvent() : I2OChain
DQMEventConsumerRegistrationInfo
«creates»
«shared_resource»
CommandQueue
+push(in stateMachineEvent)
«creates»
«creates»
«creates»
StorageManager
+receiveEventMessage()
+receiveDataMessage(in toolbox::mem::Reference)
+receiveSOAPMessage(in xoap::MessageReference)
+registerDQMEventConsumer(in in : Input, inout out : Output)
+registerEventConsumer(in in : Input, inout out : Output)
-parseDQMEventConsumerRegistration(in in : Input)
-writeDQMEventconsumerRegistrationResults(in queue_id : unsigned long, inout out : Output)
+writeEvent(in event : I2OChain, in out : Output)
DQMEventProcessor
FragmentProcessor
«creates»
«creates»
«shared_resource»
FragmentQueue
+pop() : I2OChain
+empty() : bool
«shared_resource»
DQMEventConsumerQueueCollection
StatisticsReporter
«shared_resource»
EventQueueCollection


toolbox::lang::Class
+addMethod()
+removeMethod()
+getMethod() : toolbox::lang::Method*
+getMethods() : MethodPtrVector
StateMachine
-process_event(in boost::statechart::event)
-processNextCommand()
+getCurrentState() : Operations
+processAllCommands()
+getDiskWriter() : DiskWriter
+getEventDistributor() : EventDistributor
«shared_resource»FragmentQueue
+pop() : I2OChain
+empty() : bool
I2OChain
«uses»
FragmentProcessor
+processMessages()
+updateStatistics()
+haveStateProcessFragment(in f : I2OChain)
+registerEventConsumer(in ri : EventConsumerRegistrationInfo) : QueueID
StorageManager
«uses»
«shared_resource»
CommandQueue
+push(in stateMachineEvent)
Operations
EventDistributor
FragmentStore
+addFragment(inout frag : I2OChain) : bool


I2OChain
-id
-ref
+empty() : bool
+complete() : bool
+addToChain(in f : I2OChain)
+I2OChain(in c : I2OChain)
+getBufferData() : unsigned long*
+addEventConsumerTag(in id : unsigned long)
+addStreamTag(in id : unsigned long)
+addDQMEventConsumerTag(in id : unsigned long)
+hasEventConsumerTag(in id : unsigned long) : bool
+hasStreamTag(in id : unsigned long) : bool
+hasDQMEventConsumerTag(in id : unsigned long) : bool
-streamTags
-dqmEventConsumerTags
-eventConsumerTags
«datatype»
QueueID
Configuration
+getConfiguration() : <unspecified>
+configurationHasChanged() : bool
+getActiveConsumerTimeout()

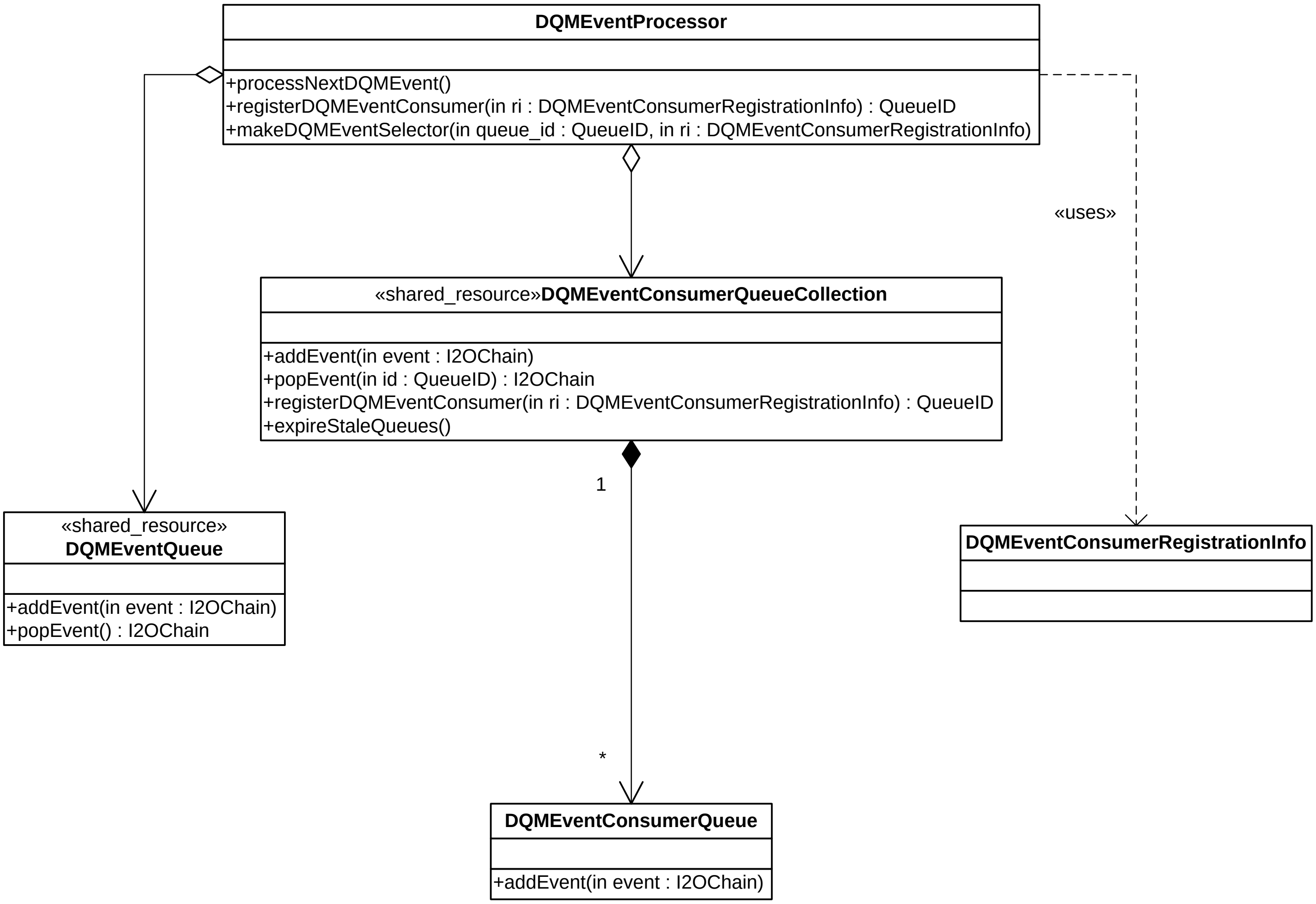


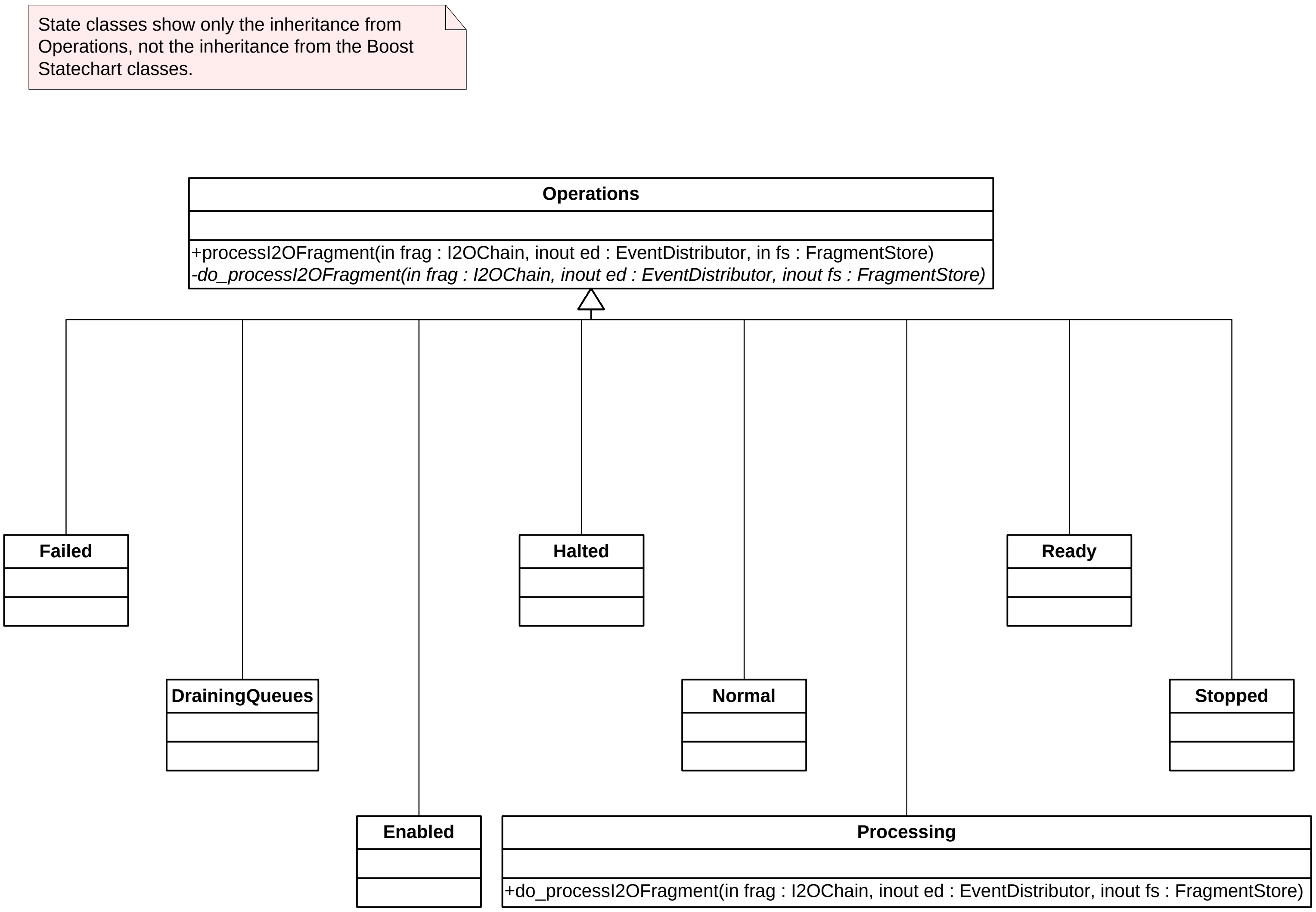
makeEventSelector uses the given EventConsumerRegistrationInfo to create a new EventSelector object; this object is used to mark events pass the selection criteria it encodes.

Responsible for tagging events for routing.

Responsible for routing events according to their tags.

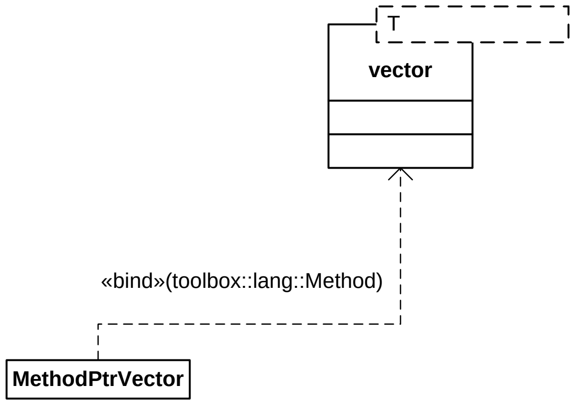
EventDistributor
+addEventToRelevantQueues(in event : I2OChain)
+full() : bool
+registerEventConsumer(in ri : EventConsumerRegistrationInfo) : QueueID
-makeEventSelector(in queue_id : QueueID, in ri : EventConsumerRegistrationInfo)
+registerEventStreams(in vector<EventStreamConfigurationInfo>)
+deactivateEventSelectors(in queue_id)
1
*
1
*
EventStreamSelector
+acceptEvent(inout event : I2OChain) : bool
EventConsumerSelector
+acceptEvent(inout event : I2OChain) : bool
«shared_resource»EventQueueCollection
+addEvent(in event : I2OChain) : void
+popEvent(in id : QueueID) : I2OChain
+popEventByConsumerID(in id : ConsumerID) : I2OChain
+clearStaleQueues() : void
+createQueue() : QueueID
1
*
EventConsumerQueue
+addEvent(in event : I2OChain)
+popEvent() : I2OChain
«shared_resource»
StreamQueue
+enq_wait(in event : I2OChain)
+popEvent() : I2OChain
«shared_resource»
DQMEventQueue
+addEvent(in event : I2OChain)
+popEvent() : I2OChain
DiskWriter
+writeNextEvent()
+configureStreams(inout vector<EventStreamConfgurationInfo>)
+makeEventStreams()
+parseEventStreamConfig()
+destroyStreams()

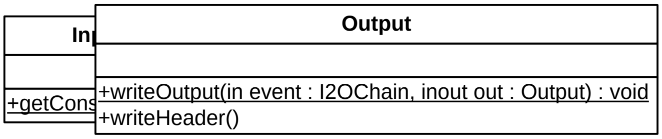

DQMEventProcessor
+processNextDQMEvent()
+registerDQMEventConsumer(in ri : DQMEventConsumerRegistrationInfo) : QueueID
+makeDQMEventSelector(in queue_id : QueueID, in ri : DQMEventConsumerRegistrationInfo)
«uses»
«shared_resource»DQMEventConsumerQueueCollection
+addEvent(in event : I2OChain)
+popEvent(in id : QueueID) : I2OChain
+registerDQMEventConsumer(in ri : DQMEventConsumerRegistrationInfo) : QueueID
+expireStaleQueues()
«shared_resource»
DQMEventQueue
+addEvent(in event : I2OChain)
+popEvent() : I2OChain
1
DQMEventConsumerRegistrationInfo
*
DQMEventConsumerQueue
+addEvent(in event : I2OChain)



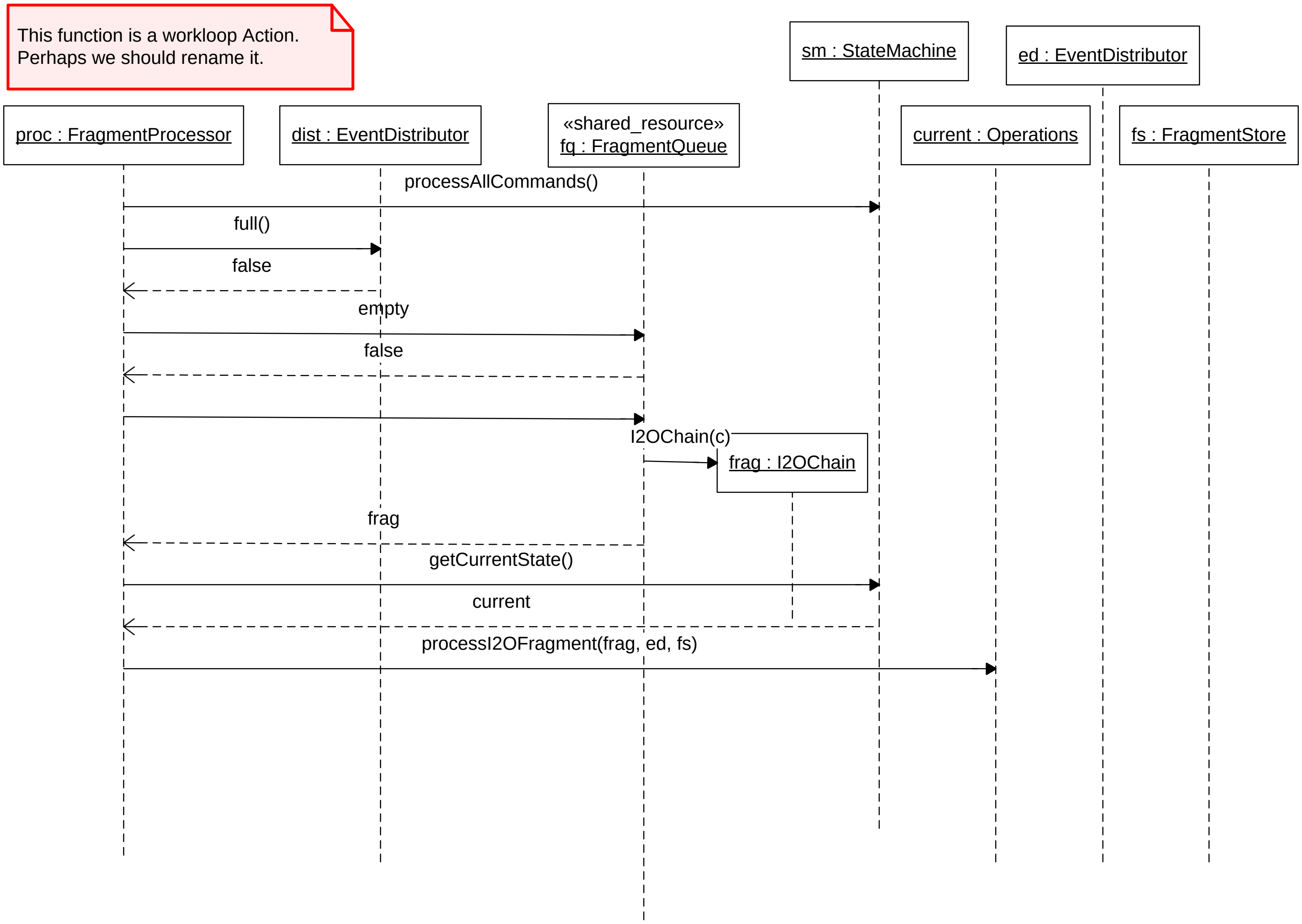
State classes show only the inheritance from Operations, not the inheritance from the Boost Statechart classes.

Operations
+processI2OFragment(in frag : I2OChain, inout ed : EventDistributor, in fs : FragmentStore)
-do_processI2OFragment(in frag : I2OChain, inout ed : EventDistributor, inout fs : FragmentStore)
Failed
Halted
Ready
DrainingQueues
Normal
Stopped
Enabled
Processing
+do_processI2OFragment(in frag : I2OChain, inout ed : EventDistributor, inout fs : FragmentStore)


T
vector
«bind»(toolbox::lang::Method)
MethodPtrVector

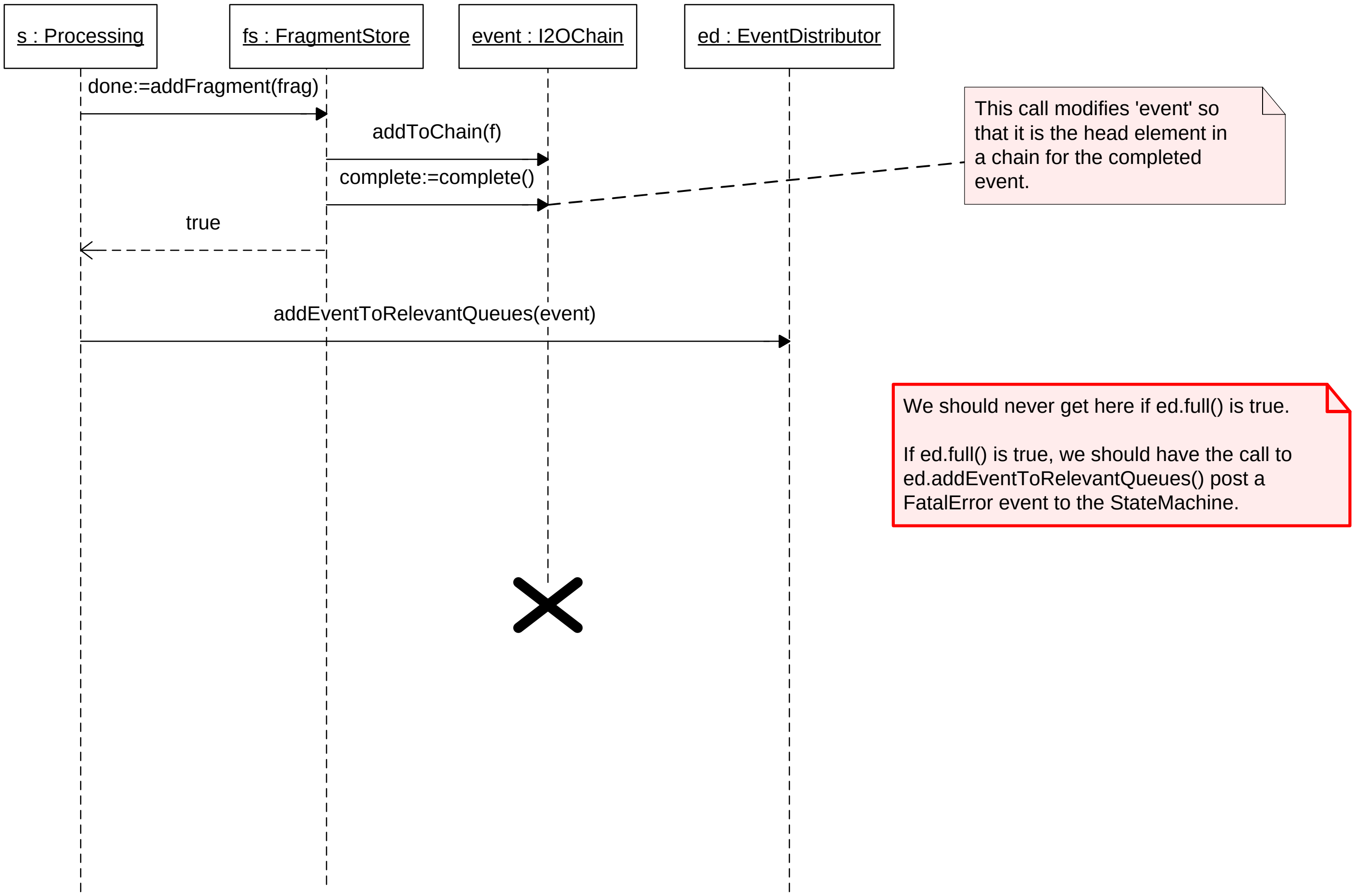

Inp
+getCons
Output
+writeOutput(in event : I2OChain, inout out : Output) : void
+writeHeader()

FragmentProcessor::processMessages



This function is a workloop Action.
Perhaps we should rename it.

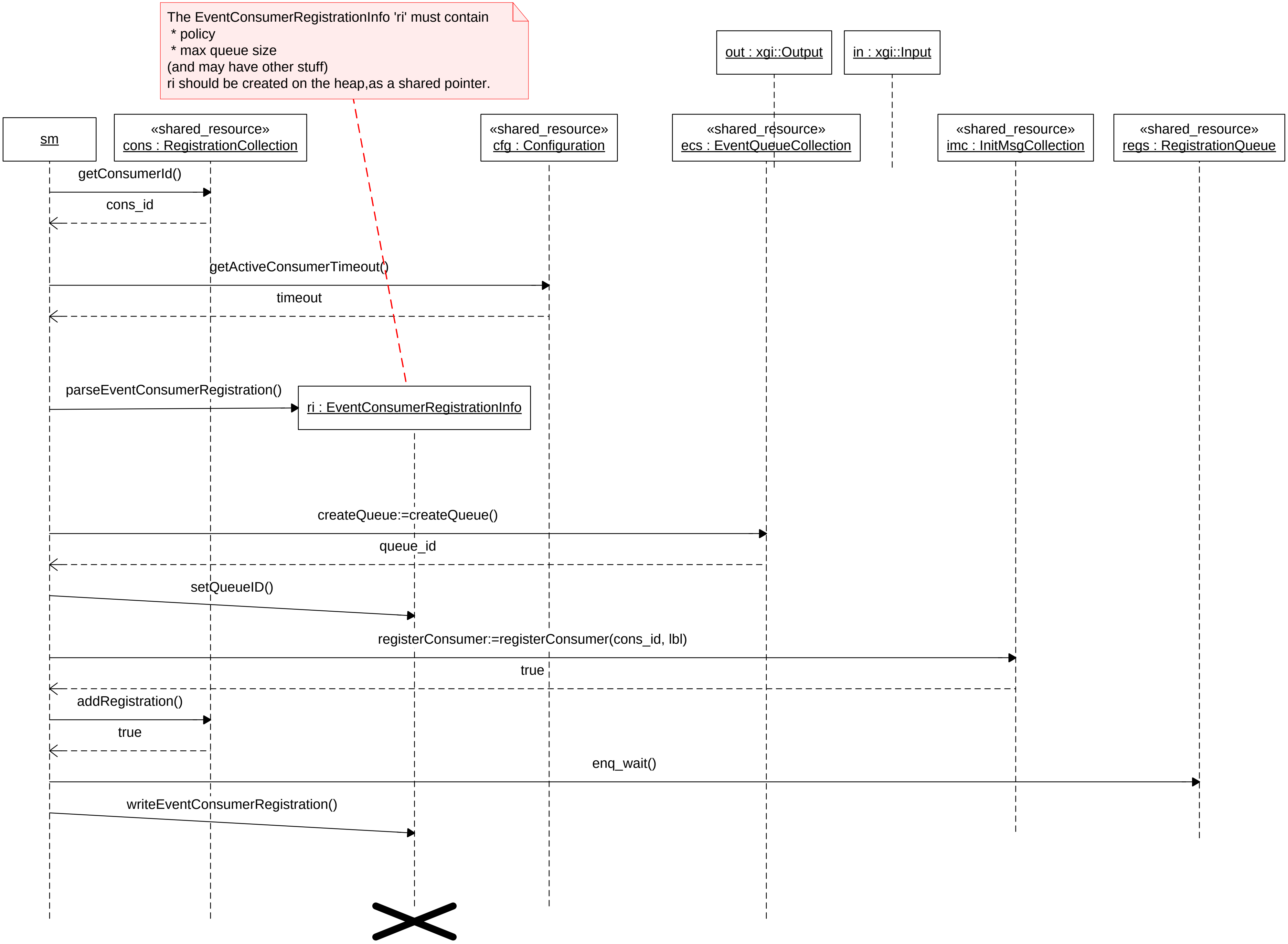
Processing State processing a fragment: last fragment case (do_processI2OFragment)



This call modifies 'event' so that it is the head element in a chain for the completed event.

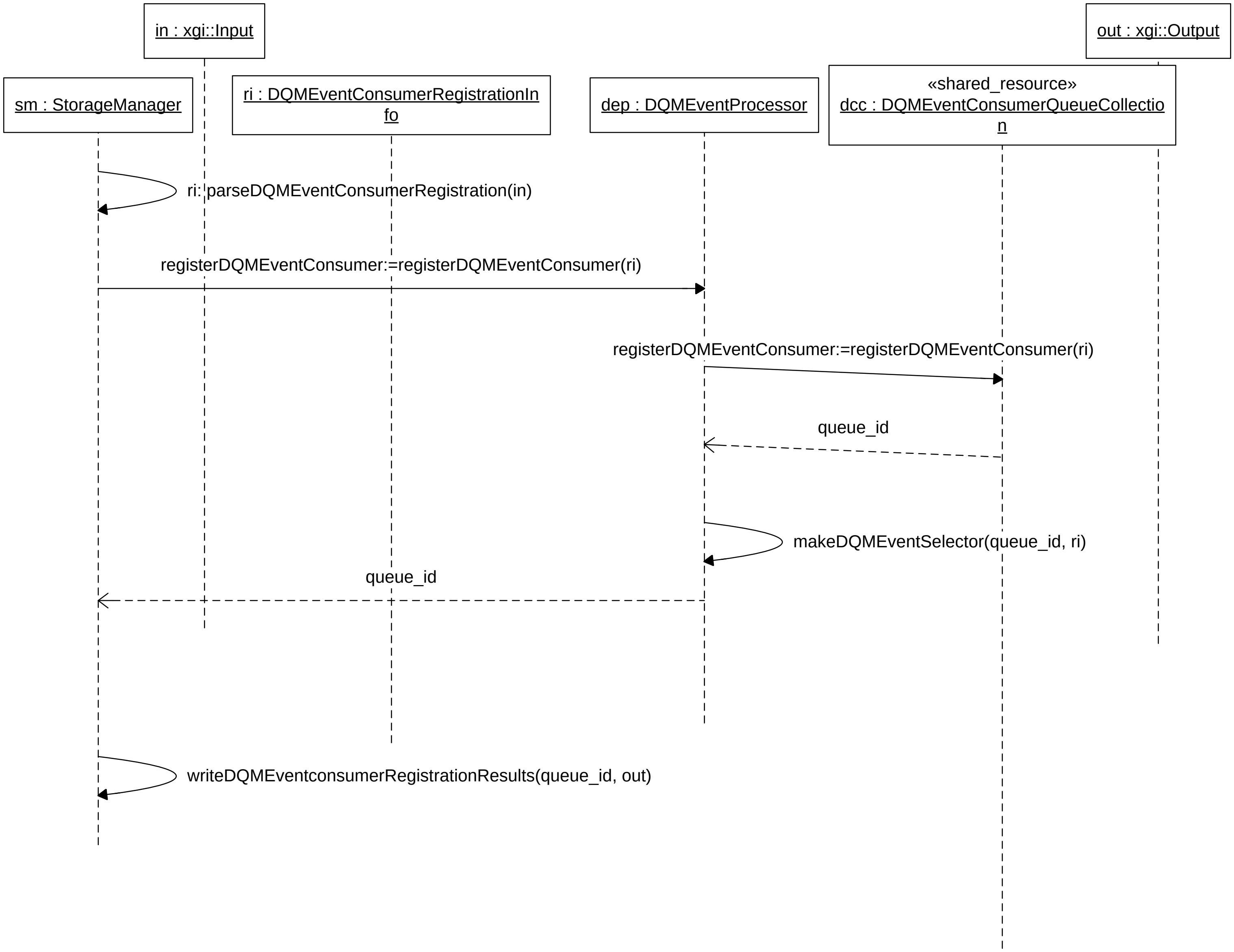
We should never get here if ed.full() is true.

If ed.full() is true, we should have the call to ed.addEventToRelevantQueues() post a FatalError event to the StateMachine.

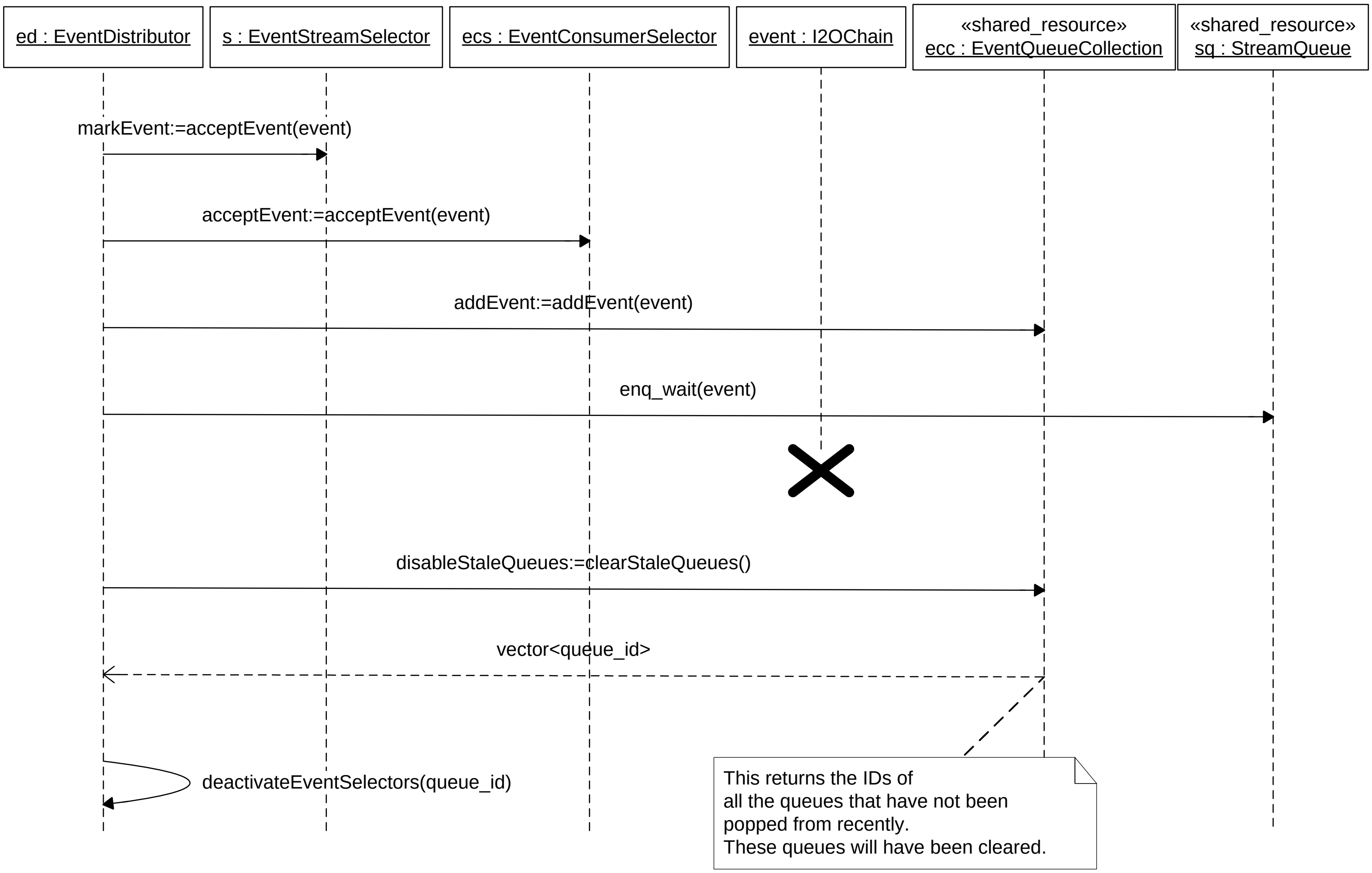


The EventConsumerRegistrationInfo 'ri' must contain
 * policy
 * max queue size
(and may have other stuff)
ri should be created on the heap,as a shared pointer.

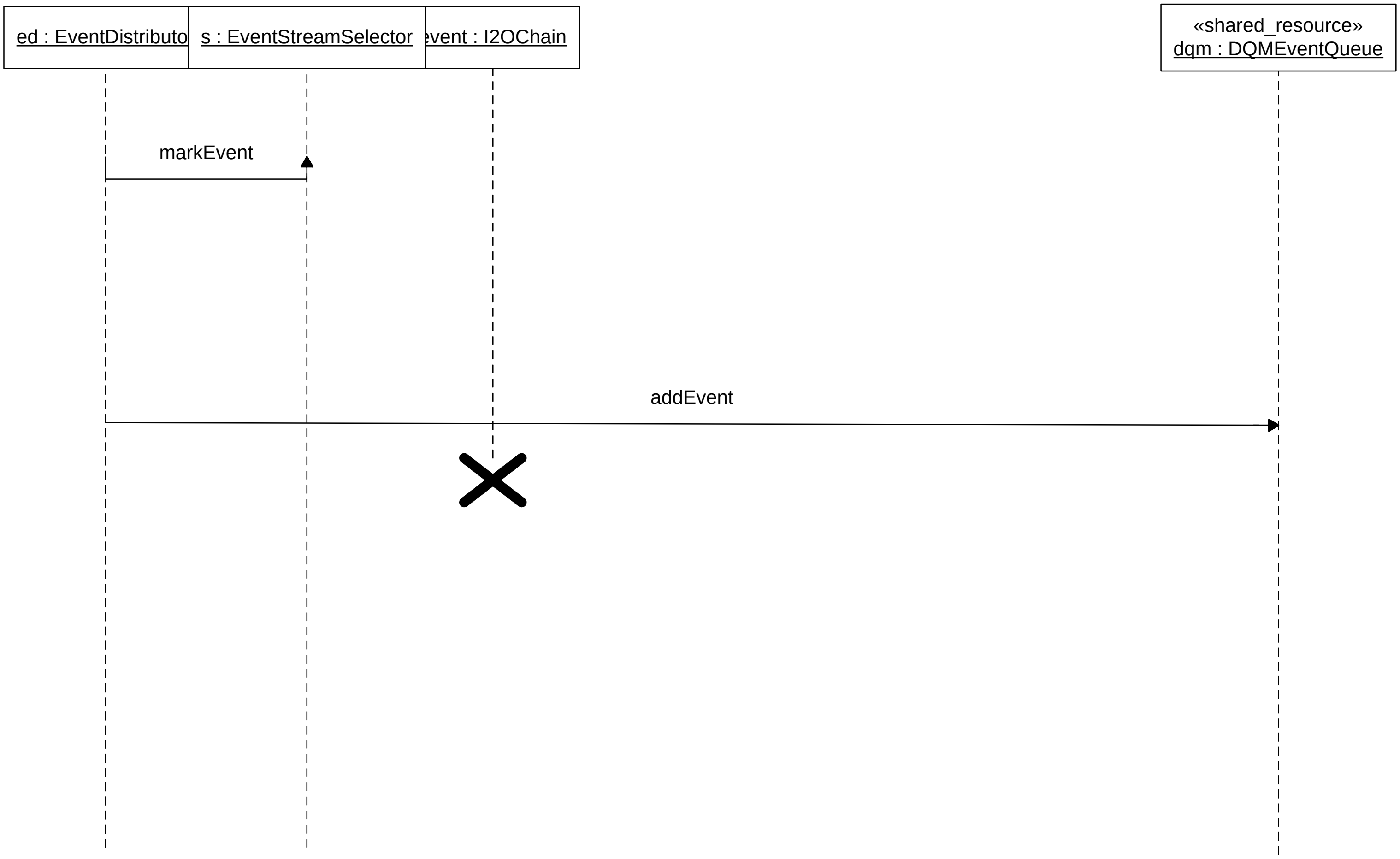
DQMEvent consumer registration (StorageManager::registerDQMEventConsumer)


in : xgi::Input
out : xgi::Output
sm : StorageManager
ri : DQMEventConsumerRegistrationInfo
dep : DQMEventProcessor
«shared_resource»
dcc : DQMEventConsumerQueueCollection
ri: parseDQMEventConsumerRegistration(in)
registerDQMEventConsumer:=registerDQMEventConsumer(ri)
registerDQMEventConsumer:=registerDQMEventConsumer(ri)
queue_id
makeDQMEventSelector(queue_id, ri)
queue_id
writeDQMEventconsumerRegistrationResults(queue_id, out)

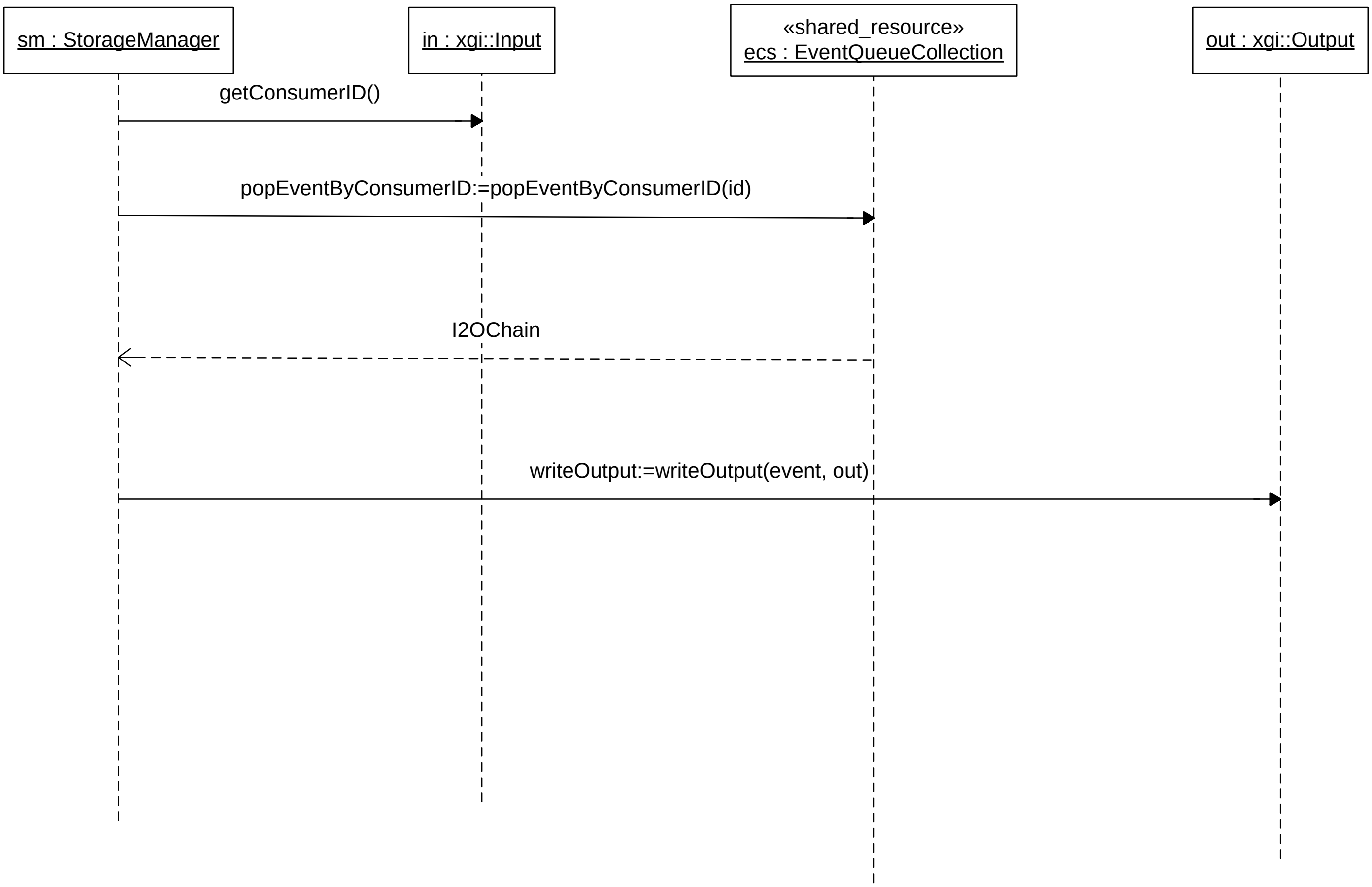
EventDistributor::addEventToRelevantQueues (collision event)


ed : EventDistributor
s : EventStreamSelector
ecs : EventConsumerSelector
event : I2OChain
«shared_resource»
ecc : EventQueueCollection
«shared_resource»
sq : StreamQueue
markEvent:=acceptEvent(event)
acceptEvent:=acceptEvent(event)
addEvent:=addEvent(event)
enq_wait(event)
disableStaleQueues:=clearStaleQueues()
vector<queue_id>
deactivateEventSelectors(queue_id)
This returns the IDs of
all the queues that have not been
popped from recently.
These queues will have been cleared.

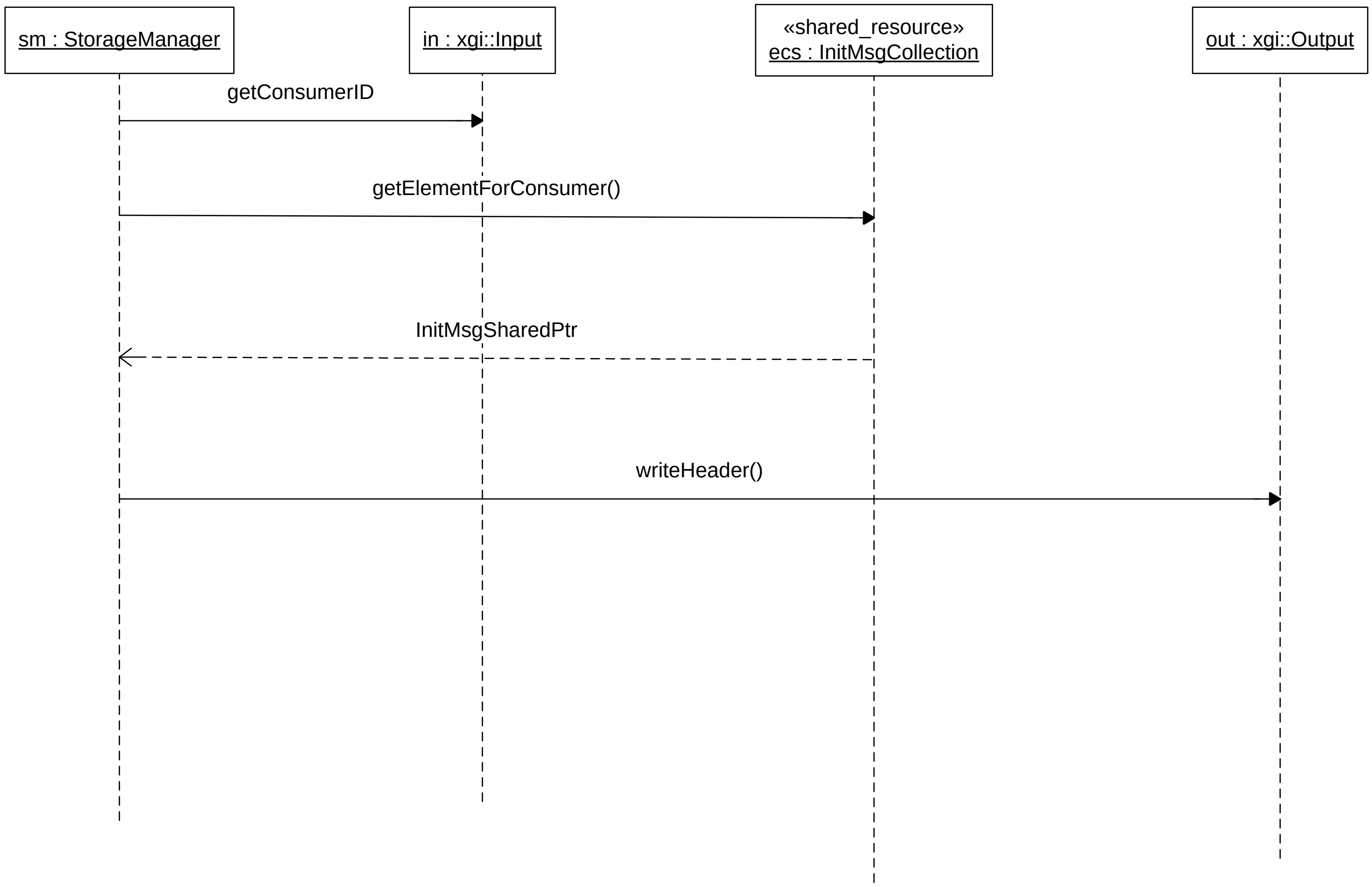
EventDistributor::addEventToRelevantQueues (DQM event)



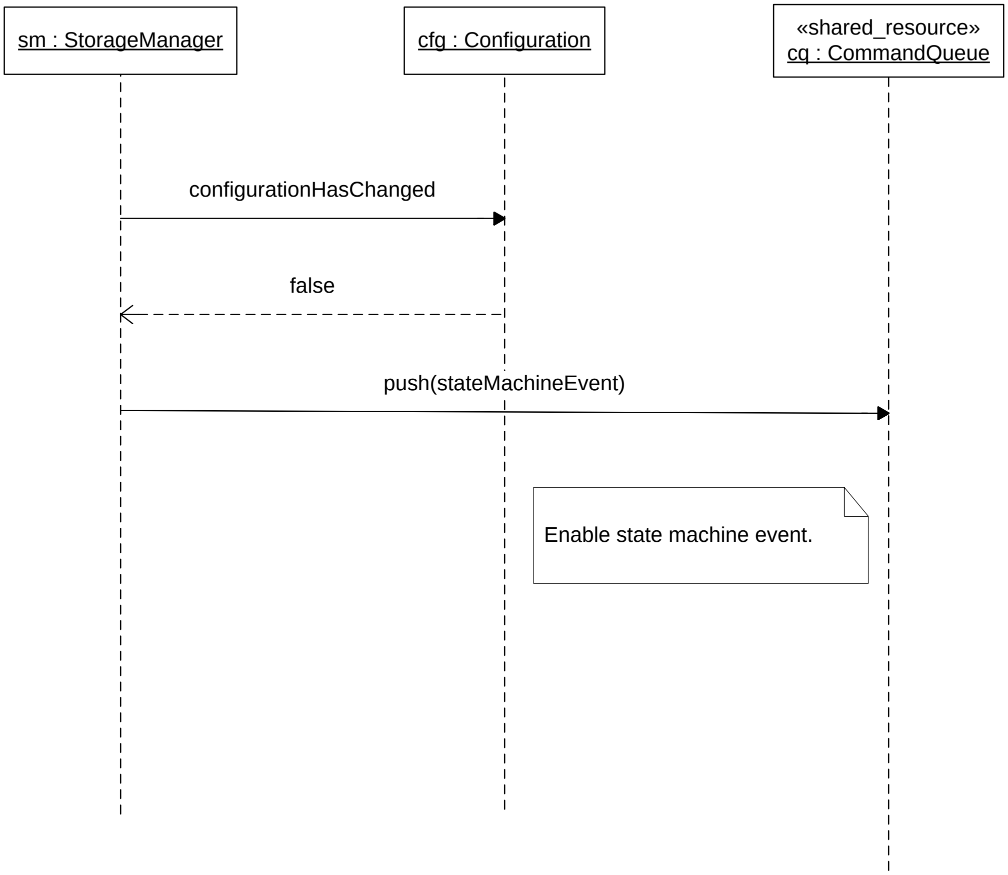
Event consumer event request (StorageManager::eventdataWebPage)


sm : StorageManager
in : xgi::Input
«shared_resource»
ecs : EventQueueCollection
out : xgi::Output
getConsumerID()
popEventByConsumerID:=popEventByConsumerID(id)
I2OChain
writeOutput:=writeOutput(event, out)

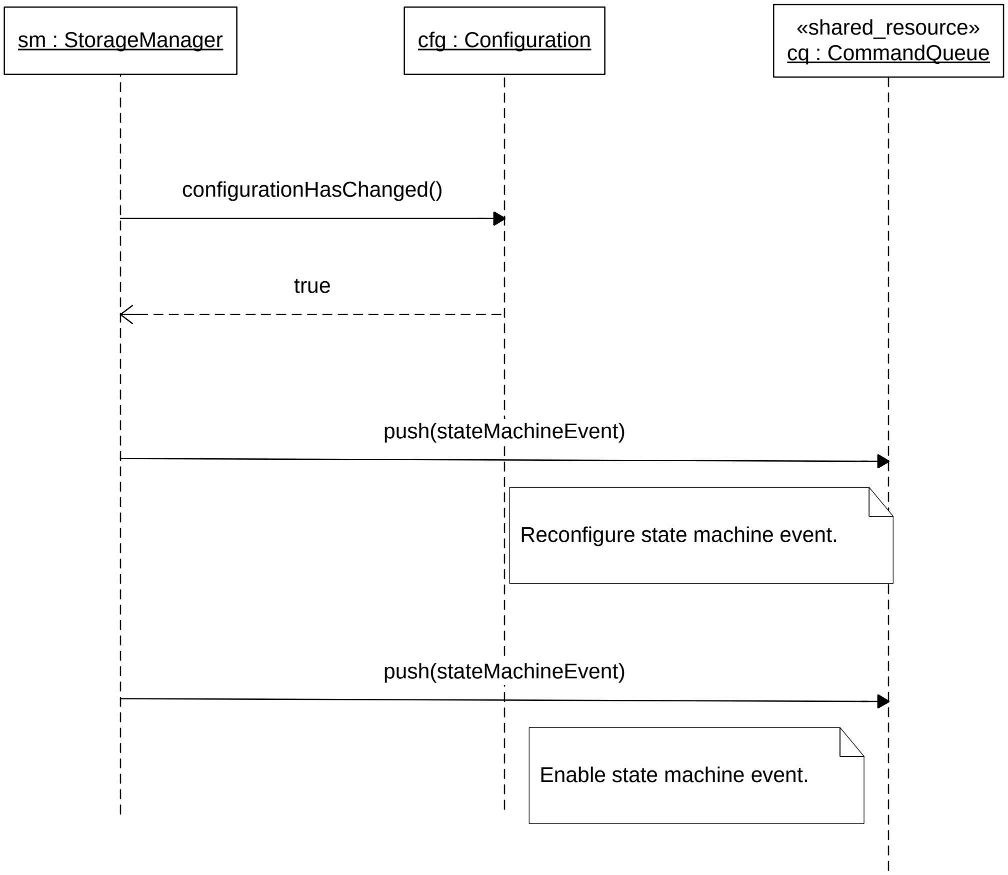
Event consumer header (INIT msg) request (StorageManager::headerdataWebPage)



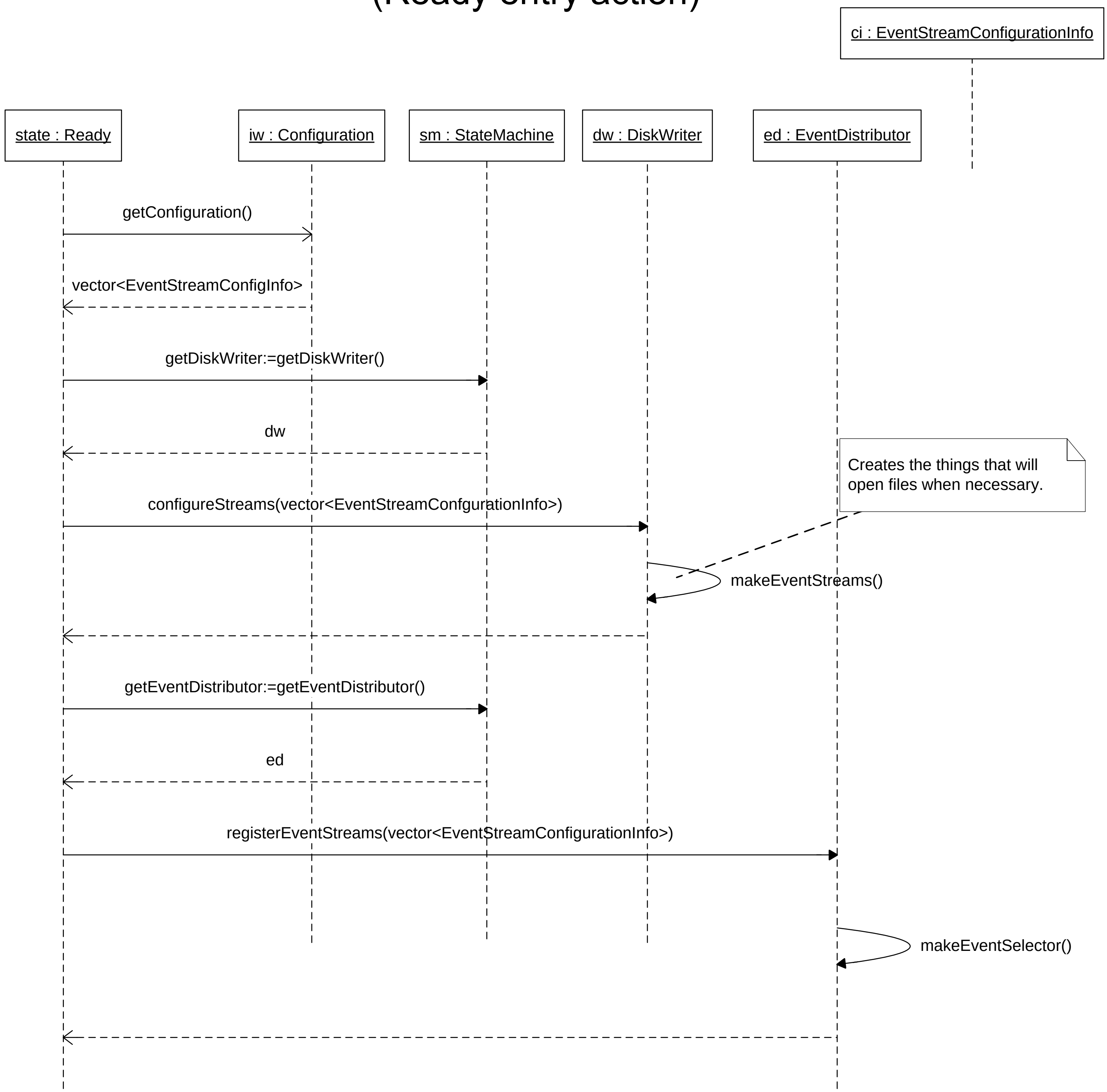
SM callback for Enable SOAP command (reconfiguration not needed)



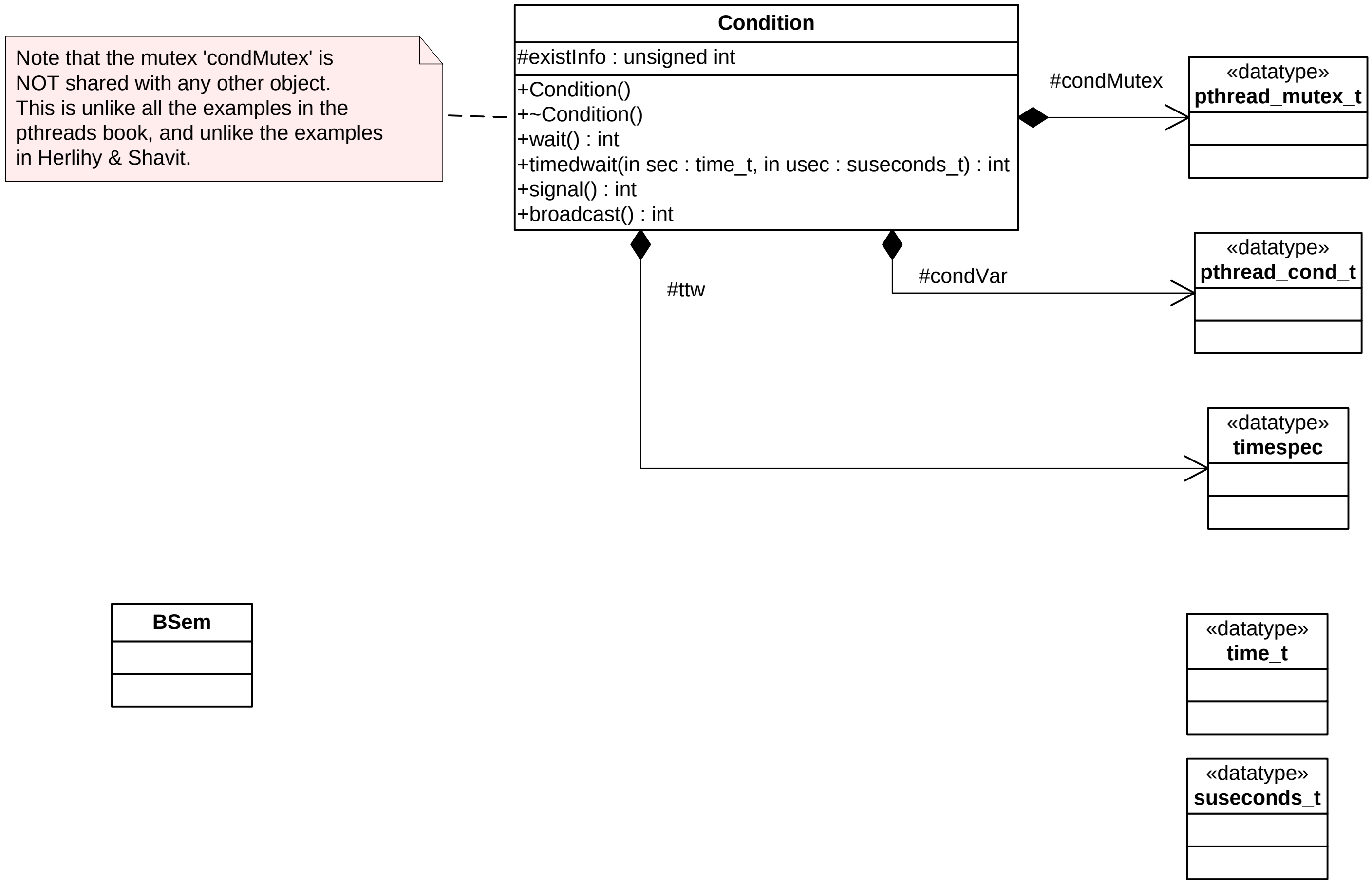
SM callback for Enable SOAP command (reconfiguration needed)



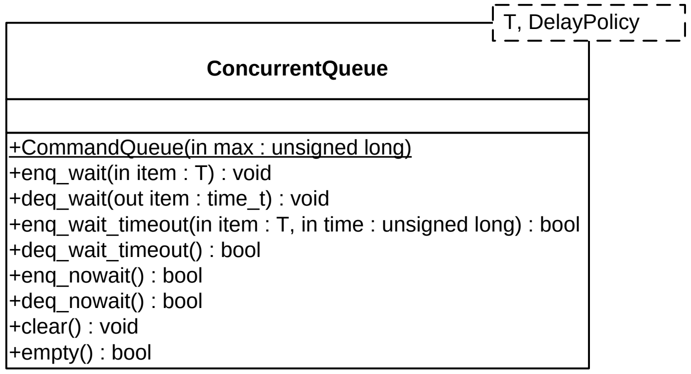
Event stream configuration
(Ready entry action)


ci : EventStreamConfigurationInfo
state : Ready
iw : Configuration
sm : StateMachine
dw : DiskWriter
ed : EventDistributor
getConfiguration()
vector<EventStreamConfigInfo>
getDiskWriter:=getDiskWriter()
dw
Creates the things that will
open files when necessary.
configureStreams(vector<EventStreamConfgurationInfo>)
makeEventStreams()
getEventDistributor:=getEventDistributor()
ed
registerEventStreams(vector<EventStreamConfigurationInfo>)
makeEventSelector()

Toolbox classes for XDAQ: Linux implementations



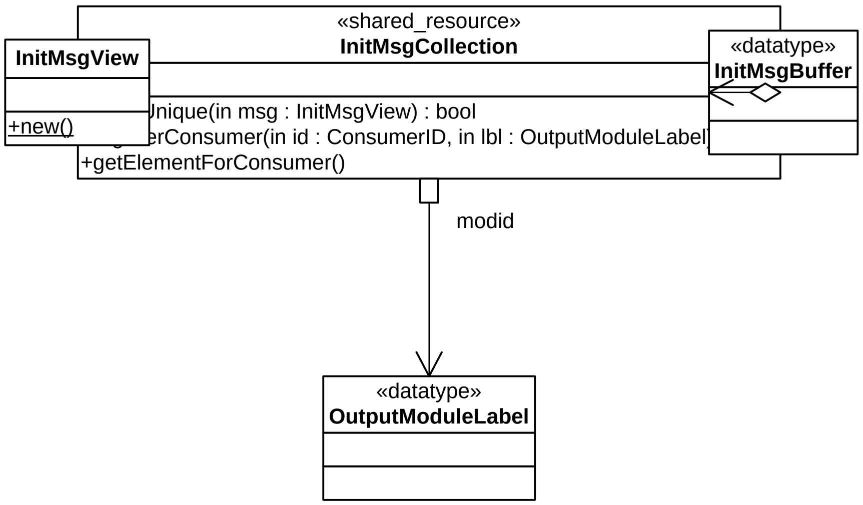
Note that the mutex 'condMutex' is NOT shared with any other object. This is unlike all the examples in the pthreads book, and unlike the examples in Herlihy & Shavit.



T, DelayPolicy

ConcurrentQueue

+CommandQueue(in max : unsigned long)
+enq_wait(in item : T) : void
+deq_wait(out item : time_t) : void
+enq_wait_timeout(in item : T, in time : unsigned long) : bool
+deq_wait_timeout() : bool
+enq_nowait() : bool
+deq_nowait() : bool
+clear() : void
+empty() : bool


«shared_resource»
InitMsgCollection
Unique(in msg : InitMsgView) : bool
erConsumer(in id : ConsumerID, in lbl : OutputModuleLabel)
+getElementForConsumer()
InitMsgView
+new()
«datatype»
InitMsgBuffer
modid
«datatype»
OutputModuleLabel