Robotics Service Bus - Enhancement #421
Use Sequence Numbers in Events (instead of full UUIDs)

07/14/2011 06:37 PM - J. Moringen

Status:	Resolved	**Start date:**	07/14/2011
Priority:	Urgent	**Due date:**	
Assignee:	S. Wrede	**% Done:**	100%
Category:	Protocol	**Estimated time:**	0.00 hour
Target version:	0.4		

Description

Proposal (original idea: Stefan Herbrechtsmeier):
Change Notification.id (which currently is a UUID) to a sequence number
Implementation:

- Change Protocol.proto
  - Change Notification.id -> Notification.sequence_number □
  - Move MetaData.sender_id -> Notification.sender_id □
- Generate sequence numbers in participants
  - Probably needs atomic integer increment in the participant - locking is probably worse than UUIDs
  - C++ (□) atomicity is currently not guaranteed
  - Python □
  - Java □
  - Common Lisp □
- Implement lazy UUID computation (e.g with v3 UUID, namespace: participant id, name: event sequence number)
  - C++ □
  - Python □
  - Java □
  - Common Lisp □
- Update [[Events|specification]] □

Open Questions:

- Which integer width is required to prevent overflows?
- Are overflows acceptable?

See: [[Meetings2011-07-14#Event-Sequence-Numbers-Stefan]]

Associated revisions

Revision 0c9dcab8 - 07/20/2011 07:03 AM - J. Moringen

Changed id to sequence number in RSBProtocol/Protocol.proto

refs #421

- RSBProtocol/Protocol.proto: change bytes Notification.id -> uint32 Notification.sequence_number

Revision a5c3932a - 07/20/2011 07:12 AM - J. Moringen

Changed events to use sequence numbers and only derive UUIDs

refs #421

- src/rsb/Event.{h,cpp}: changed Event::id into a pointer which is not set initially, but computed lazily; added Event::sequenceNumber
- src/rsb/Informer.h: set event sequence number in Informer::checkedPublish; added sequence number counter WHICH IS NOT YET THREAD-SAFE
- src/rsb/transport/spread/Assembly.{h,cpp}: use sequence numbers

instead of ids to identify corresponding events of notifications
- src/rsb/transport/spread/OutConnector.cpp: store event sequence number instead of id in notification
- src/rsb/transport/spread/ReceiverTask.cpp: extract event sequence number instead of id from notification
- test/rsb/transport/spread/AssemblyTest.cpp: adapted created of mock notifications

Revision 5e8072c8 - 07/20/2011 12:32 PM - J. Moringen

Fixed event id computation in src/rsb/Event.cpp
refs #421

- src/rsb/Event.cpp: added missing zero-padding when computing name component for UUID v5 generation

Revision 3c27217f - 07/20/2011 07:15 PM - J. Moringen

Moved sender_id to Notification in RSBProtocol/Protocol.proto
refs #421

- RSBProtocol/Protocol.proto: moved MetaData.sender_id to Notification.sender_id; sequence_number and sender_id uniquely identify the event; both are still available even if Notification.meta_data is omitted (this is relevant for fragmented events)

Revision 403b89ed - 07/21/2011 09:03 AM - J. Moringen

Adapted to changed Notification protocol buffer message definition
refs #421

- src/rsb/transport/spread/OutConnector.cpp: store sender id in directly in notification object rather than meta data object
- src/rsb/transport/spread/ReceiverTask.cpp: extract sender id from Notification object rather than meta data object

Revision f9d4be78 - 07/23/2011 11:00 PM - J. Moringen

Use sender id + seq num in src/rsb/transport/spread/Assembly.{h,cpp}
refs #421

- src/rsb/transport/spread/Assembly.{h,cpp}: use the concatenation of sender id and event sequence number as unique event identifier

Revision 6cf52a13 - 07/23/2011 11:19 PM - J. Moringen

Added sequence number type and generation in src/{types,util}.lisp
refs #421

- src/util.lisp (make-sequence-number-generator): new function; returns a function which return sequence numbers in a thread-safe way
- src/types.lisp (sequence-number): new type; 32-bit unsigned integer used for sequence numbers
- src/package.lisp (package rsb): added exported symbol sequence-number
- cl-rsb.asd (system cl-rsb): added dependency of src/util.lisp on

src/types.lisp

Revision fe660653 - 07/23/2011 11:19 PM - J. Moringen

Use sequence numbers in events; compute event ids only when needed

refs #421

- src/event.lisp (event): removed superclass uuid-mixin; updated documentation
  (event::sequence-number): new slot; stores a sequence number
  (event::id): made writer private; allow nil value; added documentation
  (shared-initialize :after event t): invalid stored id
  (event-id :before event): compute event id by calling `%maybe-set-event-id'
  (setf event-sequence-number :after t event): new method; invalidate stored id
  (setf event-origin :after t event): likewise
  (event=): changed keyword parameter compare-id? -> compare-sequence-numbers?
  (print-object event t): call `%maybe-set-event-id' to force id computation if necessary
  (%maybe-set-event-id): new function; helper function to compute event id based on origin id and sequence number
- src/informer.lisp (informer::sequence-number-generator): new slot; stores a sequence number generation function
  (send informer event): call the sequence number generator and store the resulting sequence number in the event
- src/transport/spread/conversion.lisp (one-notification->event): extract the sequence number instead of the id from the notification
  (event->notifications): store the sequence number instead of the id
  (make-notification): likewise
- cl-rsb.asd (system cl-rsb): added dependency of src/event.lisp on src/types.lisp

Revision 299abb67 - 07/23/2011 11:19 PM - J. Moringen

Adapted unit test for event changes in test/{event,package}.lisp

refs #421

- src/package.lisp (package rsb): added exported symbol event-sequence-number
- test/package.lisp (test suite root): in local function `'check-event', allow event-id to be nil
- test/event.lisp (test event-root::comparison): adapted to changed interface of `event='; construct events without delay since UUIDs are no longer involved

Revision c667a44e - 07/23/2011 11:19 PM - J. Moringen

Changed id -> seq. num. in src/transport/spread/fragmentation.lisp

refs #421

- src/transport/spread/fragmentation.lisp (assembly::id): changed type octet-vector -> sequence-number
  (print-object assembly t): print id as sequence number rather than

partial UUID
(ensure-assembly assembly-pool integer integer): changed specializer
simple-array -> integer
(merge-fragment assembly-pool t): identify the notification using
`notification-sequence-number'; `notification-id' has been removed

Revision 92d85e4a - 07/23/2011 11:19 PM - J. Moringen

Adjusted tests in test/transport/spread/fragmentation.lisp
refs #421

- test/transport/spread/fragmentation.lisp
  (test suite fragmentation-root): in local function
  `make-notification', use sequence numbers instead of ids
  (test fragmentation-root::assemble-smoke): likewise
  (test fragmentation-root::roundtrip): likewise
  (test fragmentation-root::warnings): likewise
  (test pruning-assembly-pool-root::prune): likewise

Revision edee9977 - 07/23/2011 11:19 PM - J. Moringen

Adapted sender-id handling in src/transport/spread/conversion.lisp
refs #421

- src/transport/spread/conversion.lisp (one-notification->event):
  obtain sender id from notification instance rather than contained
  meta-data instance
  (make-notification): store sender id in notification instance rather
  than contained meta-data instance

Revision 09def042 - 07/23/2011 11:19 PM - J. Moringen

Fixed sender id extraction in src/transport/spread/conversion.lisp
refs #421

- src/transport/spread/conversion.lisp (one-notification->event):
  extract sender id from notification, not meta-data

Revision d958b35c - 07/23/2011 11:19 PM - J. Moringen

Added test case event-root::id-computation in test/event.lisp
refs #421

- test/event.lisp (test event-root::id-computation): new test case;
  check computation of event ids based on sender ids and sequence
  numbers

Revision 70c0c617 - 07/26/2011 03:00 AM - J. Moringen

Initial attempt to adapt Python impl to changed protocol
refs #421, #356

- rsb/__init__.py: added sequenceNumber and senderId to Event class;
  Event.id is now read-only and lazily computed from sequenceNumber
  and senderId; removed senderId from MetaData class; added
  sequenceNumber counter to Informer class
- rsb/rsbspread/__init__.py: handle sequenceNumber and senderId
  according to above changes

- test/coretest.py: test Event and MetaData classes according to above changes
- test/eventprocessingtest.py: when creating Event instances for tests, supply sequence number and sender id
- test/rsbspreadtest.py: likewise

Revision 37849611 - 07/26/2011 03:41 AM - J. Moringen

Fixed notification ids in src/transport/spread/fragmentation.lisp
refs #421

- src/transport/spread/fragmentation.lisp (assembly::id): changed type sequence-number -> simple-array
  (print-object assembly t): print parts of the sender id and the complete sequence number
  (ensure-assembly assembly-pool simple-array integer): changed specializer integer -> simple-array
  (merge-fragment assembly-pool t): use `%make-key' to compute suitable keys for identifying notifications
  (%make-key): new function; compute a notification key from a sequence number and a sender id

Revision 018c8169 - 08/02/2011 02:19 AM - J. Moringen

Initial attempt to adapt Java implementation to changed protocol
refs #356, #421

- src/rsb/Event.java: Event.id defaults to null and is computed lazily; the id should be computed as described in https://code.cor-lab.org/projects/rsb/wiki/Events, but I could not find an UUID v5 for Java; as best-effort compromise, the sender id is used for now
- src/rsb/transport/spread/ReceiverTask.java: adapted deserialization to changed protocol
- src/rsb/transport/spread/SpreadPort.java: likewise for serialization
- test/rsb/EventTest.java: removed id-based assertions since the event id is no longer used in event comparison logic
- test/rsb/converter/ProtocolBufferConverterTest.java: removed id-based assertions since the event id is a computed attribute now
- test/rsb/filter/ScopeFilterTest.java: set sender id in Event objects to allow event id computation
- test/rsb/transport/spread/SpreadPortRoundtripTest.java: set sequence number of events instead of id
- test/rsb/transport/spread/SpreadPortTest.java: likewise

Revision 9d34baca - 08/05/2011 04:24 PM - S. Wrede

Support for name-based UUIDs according to Version 5 specifcation for EventIds (this fixes #421).
Id class refactored to EventId
Added UUIDTools class for UUID related helper functions.
Removed ControversialRules from PMD ruleset.

History

#1 - 07/15/2011 11:00 PM - J. Moringen

- Description updated

#2 - 07/18/2011 12:46 AM - J. Moringen

- Description updated

- Assignee deleted (J. Wienke)

#3 - 07/18/2011 12:46 AM - J. Moringen

- Description updated

#4 - 07/19/2011 09:22 AM - J. Moringen

- Priority changed from Low to Urgent

#5 - 07/19/2011 09:24 AM - J. Moringen

- Status changed from New to In Progress

- Assignee set to J. Moringen

#6 - 07/19/2011 10:58 AM - S. Herbrechtsmeier

Overflows shouldn`t be a problem if the message contains a CreateTime. If it can be ensured that the CreateTime always differ between two following messages from one participant the SequenceNumber can be omitted and replaced by a requirement for the CreateTime. Maybe the CreateTime should be moved from the Meta-Data to the main components of the event message.

#7 - 07/20/2011 04:19 AM - J. Moringen

- Description updated

#8 - 07/20/2011 04:25 AM - J. Moringen

We probably can't ensure strictly increasing creation times when relying on the respective operating system's clock and storing timestamps with microsecond accuracy. Using timestamps would be nice since it would avoid locking and other kinds of threading issues. I think, it's easier to go with the sequence number approach for now.

#9 - 07/20/2011 07:22 AM - J. Moringen

- Description updated

- % Done changed from 0 to 30

#10 - 07/20/2011 11:59 AM - J. Moringen

- Description updated

#11 - 07/20/2011 07:07 PM - J. Moringen

- Description updated

#12 - 07/20/2011 07:15 PM - J. Moringen

- Description updated

#13 - 07/21/2011 08:06 AM - J. Moringen

- Description updated

- % Done changed from 30 to 60

#14 - 07/25/2011 06:13 PM - S. Wrede

- Target version changed from rsb-0.10 to 0.4

#15 - 07/26/2011 03:03 AM - J. Moringen

- Description updated

- % Done changed from 60 to 80

#16 - 07/26/2011 03:04 AM - J. Moringen

- Assignee changed from J. Moringen to S. Wrede

#17 - 08/03/2011 02:01 PM - S. Wrede

- % Done changed from 80 to 90

Final changes (almost 95%) done:

- Informer now generates sequence numbers correctly upon sending of events.
- Event class has been changed to reflect the structural changes which means the senderId and sequenceNumber are now attributes of Event.
- ReceiverTask now correctly deserializes and instantiates sequence numbers.
- SequenceNumber class implements a Protocol Buffer unsigned 32bit integer type.
- Id represents now an event id, original Id was refactored to ParticipantId to make the difference clear.

What is still missing is a correct implmentation of the uuid computation from sequenceNumber and participant id. Still, r2242 reestablishes communication compatibility between trunk versions.

#18 - 08/03/2011 02:02 PM - S. Wrede

- Description updated

#19 - 08/05/2011 04:26 PM - S. Wrede

- Status changed from In Progress to Resolved

- % Done changed from 90 to 100

Feature implemented in r2261. EventId now has a getAsUUID method which computes a V5 UUID on demand if not done perviously.

This was a particularly educating issue... ;-)

#20 - 08/05/2011 04:26 PM - S. Wrede

- Description updated