Speculo

Shared memory made easy

Alberto Mardegan, Luxoft (and formerly Nokia, Canonical)

Motivation and goals

- D-Bus perceived as a bottleneck
- Alternative proposed: T-Bus
- Smaller scope: just a SHM library

Speculo: interlingua for “mirror”

- 2k LOC on top of `shm_open` and `mmap`
- Lockless
- Zerocopy
- POSIX

Design principles

- Data committed to the SHM area is immutable
- Only a few control bytes can change value
- Readers open the SHM area in read-only mode

Useful concepts

- Memory barriers:

```
atomic_store(address, value)
```

  Guarantee that once `value` becomes visible to other processes, also anything which was written to RAM before it, is visible.

Step-by-step

We are going to see how data is written to the SHM area

(the reading part is left as a mental exercise to the audience)

Writer side

```
speculo_area_new(name, flags)
```

Structure on the heap

```
<name>|<flags>.....................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=INIT.>........................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=INIT.>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>...................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>...................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
```

Structure on the heap

```
<name>|<flags>|<size>.....................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>.....................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

At this point, a manifest file gets written

- Tells the reader how to locate the SHM
- Always gets updated atomically

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
```

A reader would see that no chunks have been written

Structure on the heap

```
<name>|<flags>|<size>.............
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>..................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
```

Just initializes a struct on the stack

Structure on the heap

```
<name>|<flags>|<size>.............
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>..................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             size)
```

A new chunk ID is assigned;

the one stored in the area is incremented

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>|<chunk_id>|.......
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             size)
```

Chunk size, expiration timestamp and other few control bytes are written

A status marker is added after the end, for the next chunk

Structure on the heap

```
<name>|<flags>|<size>.............
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=UNALL>|<chunk_id>|<.....>
<chunk_size>|.......................
.....................................
.................<chunk_state=UNALL>|
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             size)
```

The state is updated

Readers are still ignoring this chunk

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=ALLOC>|<chunk_id>|<.....>
<chunk_size>|........................
.....................................
..................<chunk_state=UNALL>|
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             Size)
strcpy(ptr, “Hello world!”)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=ALLOC>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
..................<chunk_state=UNALL>|
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

```
speculo_area_new(name, flags)
speculo_area_set_max_size(area, size)
speculo_area_create(area)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             Size)
strcpy(ptr, “Hello world!”)
speculo_chunk_commit(&chunk)
```

The chunk becomes visible to the readers

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
..................<chunk_state=UNALL>|
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
.....................................
```

Writer side

Writing one more chunk

```
ptr = speculo_chunk_allocate(&chunk,
                             Size)
strcpy(ptr, “Hello world!”)
speculo_chunk_commit(&chunk)
speculo_area_create_chunk(area,
                          &chunk)
ptr = speculo_chunk_allocate(&chunk,
                             Size)
strcpy(ptr, “Once upon a time...”)
speculo_chunk_commit(&chunk)
```

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
.................<chunk_state=WRITN>|
<chunk_id>|<...><chunk_size>|Once upon
a time someone started to write a nice
story, but then something happened and
the need to justify the text perfectly
turned the story a bit meaningless, or
at least not as interesting as someone
hoped it would have been but who cares
<chunk_state=UNALL>|.................
.....................................
.....................................
.....................................
.....................................
```

Writer side

When there's not enough room for writing the next chunk, a new SHM area gets allocated.

- Valid chunks get copied over
- Invalid data:
  - Expired chunks (by timestamp)
  - Obsoleted chunks (will explain later)

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
..................<chunk_state=WRITN>|
<chunk_id>|<...><chunk_size>|Once upon
a time someone started to write a nice
story, but then something happened and
the need to justify the text perfectly
turned the story a bit meaningless, or
at least not as interesting as someone
hoped it would have been but who cares
<chunk_state=UNALL>|.................
.....................................
.....................................
.....................................
.....................................
```

Writer side

When there's not enough room for writing the next chunk, a new SHM area gets allocated.

- Valid chunks get copied over
- Invalid data:
  - Expired chunks (by timestamp)
  - Obsoleted chunks (will explain later)
- Manifest file is updated to point to the new SHM

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=READY>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
.................<chunk_state=WRITN>|
<chunk_id>|<...><chunk_size>|Once upon
a time someone started to write a nice
story, but then something happened and
the need to justify the text perfectly
turned the story a bit meaningless, or
at least not as interesting as someone
hoped it would have been but who cares
<chunk_state=UNALL>|.................
.....................................
.....................................
.....................................
.....................................
```

Writer side

When there's not enough room for writing the next chunk, a new SHM area gets allocated.

- Valid chunks get copied over
- Invalid data:
  - Expired chunks (by timestamp)
  - Obsoleted chunks (will explain later)
- Manifest file is updated to point to the new SHM
- Old memory area is marked as obsolete

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=OBSOL>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
.................<chunk_state=WRITN>|
<chunk_id>|<...><chunk_size>|Once upon
a time someone started to write a nice
story, but then something happened and
the need to justify the text perfectly
turned the story a bit meaningless, or
at least not as interesting as someone
hoped it would have been but who cares
<chunk_state=UNALL>|.................
.....................................
.....................................
.....................................
.....................................
```

Writer side

When there's not enough room for writing the next chunk, a new SHM area gets allocated.

- Valid chunks get copied over
- Invalid data:
  - Expired chunks (by timestamp)
  - Obsoleted chunks (will explain later)
- Manifest file is updated to point to the new SHM
- Old memory area is marked as obsolete
- Old memory area is unlinked (current readers can still read it)

Structure on the heap

```
<name>|<flags>|<size>.................
.....................................
.....................................
.....................................
```

Structure on the SHM area

```
<state=OBSOL>|<size>|<id>|<chunk_id>|.
<chunk_state=WRITN>|<chunk_id>|<.....>
<chunk_size>|Hello world!\0...........
.....................................
.................<chunk_state=WRITN>|
<chunk_id>|<...><chunk_size>|Once upon
a time someone started to write a nice
story, but then something happened and
the need to justify the text perfectly
turned the story a bit meaningless, or
at least not as interesting as someone
hoped it would have been but who cares
<chunk_state=UNALL>|.................
.....................................
.....................................
.....................................
.....................................
```

Obsoleted data

- Chunks which have expired (by timestamp)
- Chunks which have been updated:

```
ptr = speculo_chunk_update(&chunk, size)
```

allocates a new chunk, but keeps the same ID: readers will only consider the new instance

Speculo use cases

1) Stream of messages
   - Set an expiration time on each message
   - Each message sender is a writer

2) Publishing a data block
   - Data can be updated

Question time

Code: http://gitlab.org/mardy/speculo

Docs: http://mardy.gitlab.io/speculo/

Me:

E-mail: mardy@users.sourceforge.net

Blog: http://blog.mardy.it (English, Interlingua, Italian)

http://mardy.livejournal.com (in Russian)

Twitter: mardy78