Universitat de les Illes Balears

Dynamic FT for FTTRS

Manuel Barranco

kick-off meeting of the DFT4FTT project

Friday, 21st October, 2016

project tasks

- T2. Tolerating transient faults in the async. traffic
- T3. Guaranteeing data consistency for the async. traffic
- T4. Making network-level FT and data consistency mechanisms dynamic

what are the objectives of these tasks?

what are the objectives of these tasks?

FT4FTT mostly focused on “synchronous” part of FTT

now we want to focus on

the asynchronous part and on making FT dynamic

objective of the just-mentioned tasks can be summarized as

extend FT4FTT with **mechanisms** to provide

communication services adequate

for distributed highly reliable and flexible **apps**

- we could specify the following 3 **top-down levels** of communication **requirements**:
  - communication **services** must support **app-level mechanisms** that apps use for providing a highly-reliable and adaptive **app service**
  - communication **services** must be dependable and flexible
  - communication **infrastructure** must be dependable (and flexible?)

we need to carry out the following **ideal steps**

(not necessarily sequentially)

ideal steps

- identify
  - FT and RT/FT flexibility **features apps** must exhibit
  - RT and FT communication **services to support** these app **features**
  - **existing** communication services of FTT and FT4FTT
  - **which** of those **existing** communication services are **needed**
  - **non-existing but needed** communication services
  - **existing** communication **mechanisms** and protocols of FTT and FT4FTT that may support the needed communication services

ideal steps

- design the middleware architecture to:
  - support the execution of replicas apps
  - **decompose** the **implementation** of the **communication services** in the **appropriate levels of abstraction** or modules
- **modify** existing **or design** new communication **mechanisms** and protocols as needed
- identify priorities and dependencies

Julián sketched most FT4FTT mechanisms

(mostly based on synchronous traffic)

thus

let us discuss about

some **existing communication** mechanisms and services that **mostly** rely on **asynchronous** traffic

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

non-exhaustive list

- FTT-control
  - **PnP**
  - stream registration
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

PnP

- used for registering
  - slaves (master assigns node ID)
  - apps (master assigns app ID)
- controversial mechanism from reliability point of view
  - who is authorized to trigger a PnP procedure?
  - unreliable message transmission
  - message loss/delay can cause inconsistencies
- so far deemed as an undesired mechanism

PnP

however

PnP

- we still do not have a system start-up protocol
  - may we adapt PnP for this?
- can PnP be adapted for providing flexible FT?
  - support app migration from node to node?
  - support app creation at runtime?
  - support app deletion at runtime?
- it uses a heartbeat to unregister crashed nodes
  - do PGs currently implement a similar mechanism?

non-exhaustive list

- FTT-control
  - PnP
  - **stream registration**
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

stream registration

- provides
  - negotiated stream creation (triggered by slave *manager* task)
  - endpoint registration
    - triggered by one slave producer task
    - triggered by N slave consumer tasks
  - creation of multicast group at Ethernet level (IGMP)
    - to efficiently address the slaves where prod./consumers are
  - simultaneously unblock of prod/consum. tasks

stream registration

open issues

stream registration

- can a task register and endpoint later on ??
- controversial mechanism from reliability point of view
    - who is authorized to trigger each one of the actions of an stream registration?
    - unreliable message transmission
    - message loss/delay can cause inconsistencies
    - IGMP is not reliable and does not enforce consistency
- can it be deemed as an undesirable communication service?
    - to discard it seems to limit flexibility

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - **stream properties changes**
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

stream properties changes

- provides
  - negotiated stream modification
    - triggered by one slave QoS manager task
    - triggered by master QoS manager
  - synchronized update of the SRDB and NRDB
    - to provide an smooth and consistent transition
    - based on the “stream tagging mechanism”

stream properties changes

open issues

stream properties changes

- controversial mechanism from reliability point of view
  - which slaves (if any) are authorized to request a change?
  - unreliable messages transmission
  - message loss/delay can cause inconsistency
  - does the tagging mechanism really enforce consistency even if all messages are correctly tx/rx?
- can it be deemed as an undesirable communication service?
  - to discard it seems to limit flexibility

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - **admission control**
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

admission control

- is the FTT admission control paradigm the most appropriate one?
- actions and consequences upon an admission control denial?
- maybe it should consider different levels of schedule?
  - dataRT & NRT messages
  - **FTT-control messages** (now they may be replicated)
  - new replicated messages introduced by FT4FTT
  - new potential reconfiguration messages introduced by DFT4FTT
  - actions to change FT and RT properties
    - can FT changes be somehow predicted in advance?
    - do FT changes have to be made following a given order to ensure timeliness?

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - admission control
- **FTT asynchronous RT data traffic**
- FTT NRT data traffic
- FT4FTT CVEP
- Periodic servers

FTT async. RT data traffic

- FT and DFT mechanisms needed here?
  - preventive retranmissions to attain high reliability?
  - communication service with confirmation?
  - others?

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- **FTT NRT data traffic**
- FT4FTT CVEP
- Periodic servers

FTT NRT data traffic

- can NRT data traffic be critical?
- if so, what FT and DFT mechanisms needed here?
  - preventive retranmissions to attain high reliability?
  - communication service with confirmation?
  - others?

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- **FT4FTT CVEP**
- Periodic servers

FT4FTT CVEP async. traffic

- CVEP: a “reliable communication service with multicast confirmation”
  - multiple preventive retx of each **cc-vector** in **sync**. win.
  - multiple preventive retx of each **ACK** in **async**. win.
  - each replica is reliably informed about which cc-vectors were acknowledged by which replicas

FT4FTT CVEP async. traffic

open issues

FT4FTT CVEP async. traffic

- the stream model must be adapted to cope with this type of closely-related streams
  - “replicated streams” vs “EC-synchronized multipublisher streams”
    - the former seems to be a better concept
    - the second is easier to implement
- how to consider this or similar types of streams in the schedulability analysis (admission control)?

non-exhaustive list

- FTT-control
  - PnP
  - stream registration
  - stream properties changes
  - admission control
- FTT asynchronous RT data traffic
- FTT NRT data traffic
- FT4FTT CVEP
- **Periodic servers**

periodic servers

- provide bandwidth guarantees and isolation of RT and NRT asynchronous data messages
- open issues
  - o do they serve FTT-control traffic?
  - o do they cope the whole async window?
  - o if so, how difficult would be to schedule the following traffic?
    - replicated FTT-control traffic
    - replicated RT/NRT async traffic
    - new FT4FTT/DFT4FTT replicated traffic