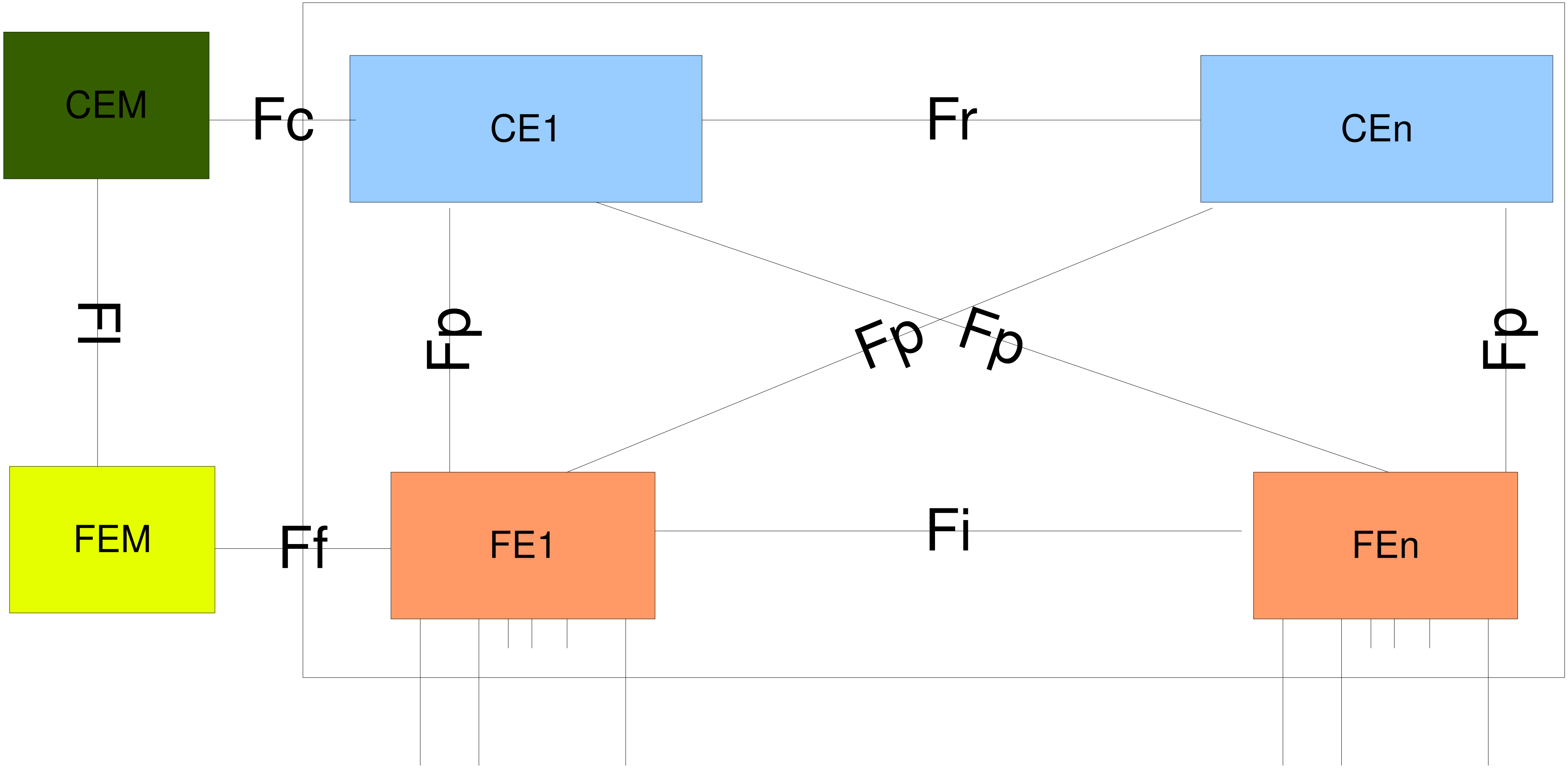
CE CE Strawman discussion

Kentaro Ogawa

Focus: *Fr* Plane



Desires

- KISS principle whenever possible
  - Not very ambitious on CE set
    - 1 Active and N backup CEs at a time
      - As defined in protocol document section 8
  - Avoid changes to ForCES architecture
    - Unchanged protocol
      - Use existing protocol constructs such as transactions
    - Unchanged model
      - Any CE configuration/state to be held in an LFB
        - Define a CE Object LFB if needed

What happens of CECE plane?

- Updates of NE state and config to backup CEs from master CE
- Fault detection by backup CEs in case master CE goes down
  - Election to select new master CE

CE Object LFB

- Store operational config and state of Fr plane
  - The NE CE set, for each CE
    - What type (eg master/slave)
    - Status (connected etc)
  - Connectivity parameters
    - Dead intervals etc
    - Do we need a CE Protocol Object LFB for this?

Operational Approach

- Each CE on bootup knows the NE CE set
- Each backup CE associates to listed master CE
- Master CE updates backup CEs with config

CE set discovery alternatives

- Simple approach
  - Retrieve the CE state and types from CEM interface
    - Very static CE list (including initial master)
- Slightly complex approach
  - Bootstrap as in simple approach above
  - Allow master CE to update CE Object of backup CEs with any other CEs it knows of
    - Backup CE connects to master CE
- Preference is for simple approach for now

FE participation

- Defined in protocol draft section 8
  - Fault detection and recovery
- We have a set of CEs to which an FE connects
- An FE associates to all CEs
  - Slightly different from what is defined in section 8
- An FE is dumb
  - It responds to any CE that requests it to do anything
- Events and redirects are sent to only listed Master CE
  - Alternate: send to CEID ALLCES

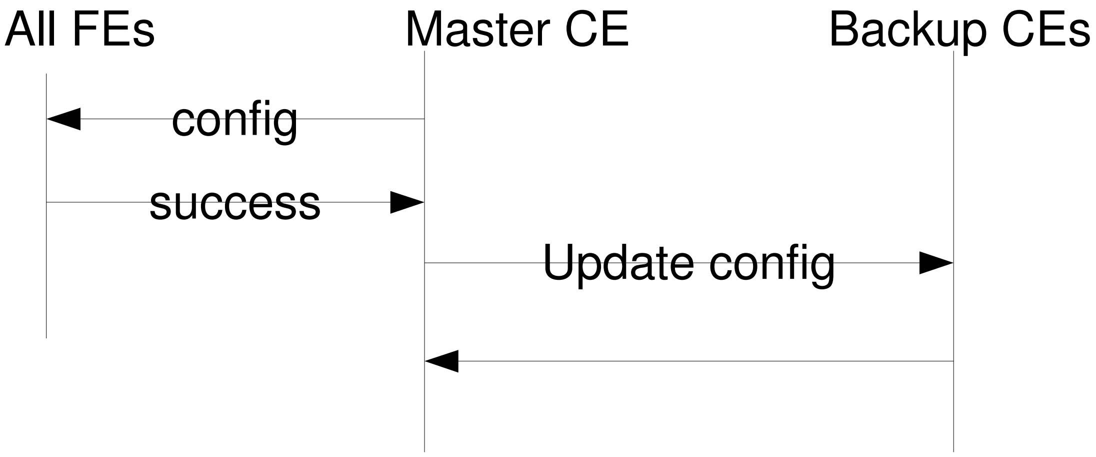
CE master election

- Very simple and static
    - The lowest CEID wins
- If master CE dies
    - All CEs associate to the next lowest CEID
        - Easy since the static list never changes

Challenges on CECE

- Master CE update/sync of backup CE
  - Async vs sync updates
- Protocol referencing affected LFB component to backup CEs
- Which CE associates to what CE?
- Avoiding split brain

Challenge: CE update sync



All FEs

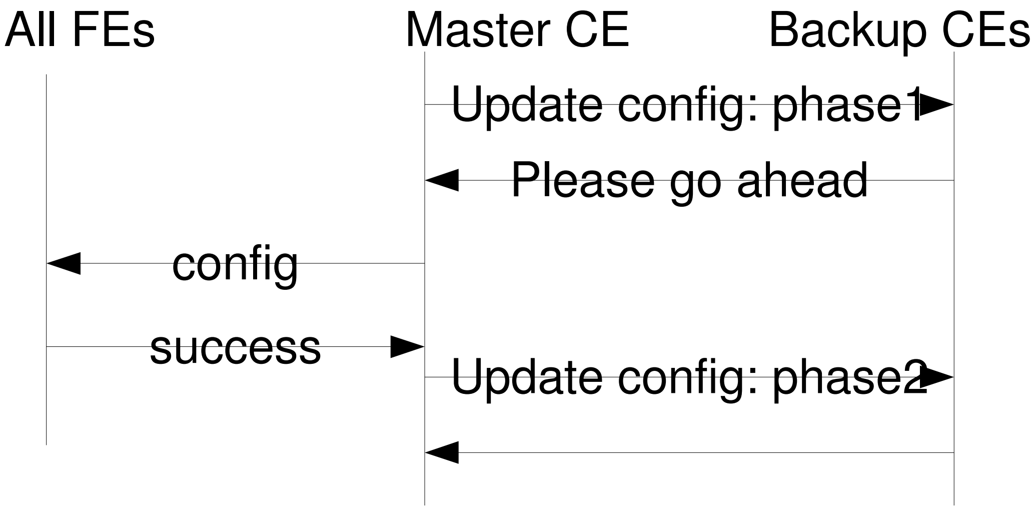
Master CE

Backup CEs

config

success

Update config



All FEs

Master CE

Backup CEs

Update config: phase1

Please go ahead

config

success

Update config: phase2

Advantage:

* no update to FE if backup CEs cant take over

--> Use PL transactional operations

Disadvantage:

* more messages exchanged per config

Challenges: Referencing update component

- FE-w/LFB-x/instance-y/component path-z is unique NE-wide
- Direction is from CEID
  - Therefore, config operation applies to hierachy:
    - FE-w/LFB-x/instance-y/component path-z
- No clean message to CEn from CE1 “this is a config set on FE-w/LFB-x/instance-y/component path-z”
  - Hierachy in message header is between two points

Solutions: Referencing update component

- CEs keep a translation table for re-mapping
    - FE-w/LFB-x/instance-y/component path-z to something they negotiate and store in CEObject
    - So then message from CE1 -> CEn translates
        - Dst = FE-w/LFB-x/instance-y/component path-z
        - to: dst = CEn/LFB-x/*instance-y`*/component path-z
    - Limits use of LFB instances
    - Adds complexity of maintaining a map

Solutions: Referencing update component

- Use multicast IDs to map the FE to which it applies to
  - Update message to CE is sent to multicast address + FEID
    - Eg FEID 1 becomes 0xC000001
  - Limits the total number of FEs in an NE to about 2^30 – 16
    - 16 less than what we specify as upper bound
  - Limits the use of multicast ID space in case needed for other things

Solutions: Referencing update component

- Introduce a new TLV at the same hierachy level as LFB selector
  - Call it “applies to” it will encompass the FEID on which update happens
    - Message now is from CEIDx to CEIDy “applies to” FEIDz on LFB-a/instance-b/path-c
  - This seems to be the cleanest solution but requires a small change to add a new TLV

CECE association

- Simple approach is that each CE associates to the known master
  - Avoids too many connections
- Upon failure of known master, election process is simple
  - Connect to next lowest CEID
  - Optimize
    - Master CE always updates the CEObject Ces table of all backup Ces with connection status of each CE

CE master split brain

- If master CE dies
  - All CEs associate to the next lowest CEID
    - Repeat until success
  - Possible that some CEs may only be able to connect to others
    - We need to make sure we survive in such a scenario