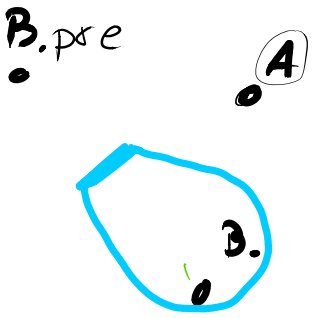
den 30 september 2015 16:14




```
type DHTMsg struct {
    Key string `json:"key"`
    Src string `json:"src"`
    Dst string `json:"dst"`
    // Other?
    Req  string `json:"req"`
    Opt  string `json:"opt"`
    Data string `json:"data"`
    //Origin string `json:"origin"`
    //Bytes string `json:"bytes"`
}
```

```
type DHTNode struct {
    nodeId      string
    successor   *DHTNode
    predecessor *DHTNode
    // Added manually:
    fingers     [bits]*DHTNode
    bindAdress string
    queue       chan *DHTMsg
    datanågot
}
```

Fully functional

Virtual nodes, only
- NodeID
- Bindadress

Objective 2

- Join
  Key : Origin Node Id
  Src: Src from the node which last sent message
  Req: "join"
  - Response/Request?
- Lookup
  - Data ?
  - Response/Request?
- Update (Predecessor and ?Successor)
  - Data ?
  - Response/Request?
- Stabilize
  - Data ?
  - Response/Request?
- SetupFingers (Update a nodes fingers)
  - Data?
  - Response?
  - Response Handle?

- dataPull
- dataPush
- dataUpdate
- nodeExit (clean) annars hanteras det som en crash.

Join: (2 fall)

- Basfall (Nod B är ej i en ring)
  - Nod A skickar join till Nod B
  - Nod B fick en join, har ingen Pre/Suc. Lägger till Nod A som Pre/Suc och skickar tillbaka en join.
  - Nod A får en join, repeterar ovanstående steg. Sen har vi en ring.
- Specialfall: (Node B är med i en ring)
  - Nod A skickar en join till nod B
  - Nod B kollar **om nod A ska vara mellan** NodB och NodB.pre.
    - if(True):
      NodB säger till nod A att dess successor är **NodB**, och att NodB.predecessor är NodAs nya predecessor.
      Säg även till **NodB.predecessor** att dess successor är **NodA**
      Efter det är gjort, sätter NodB sin nya predeccessor till Nod A.
      Lägg in en "Setup fingers" i sin queue
    - If(False):
      - ***Kör en Lookup (O(Log(n) + O(Stabilize))**
      - Skicka vidare meddelandet till sin successor ? (O(N-1))