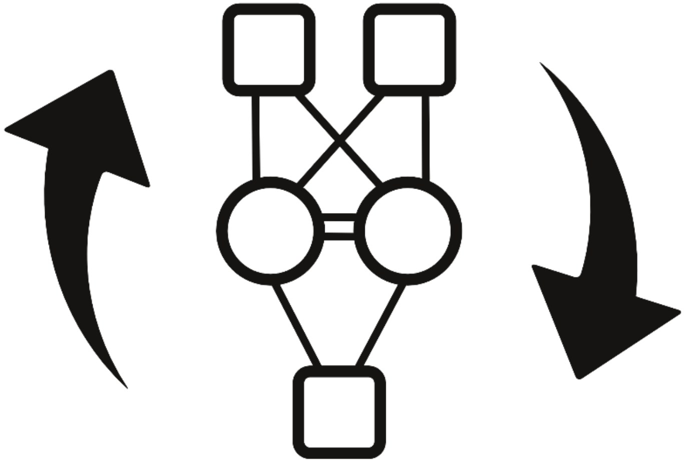
Dynamic Node Replication in the DFT4FTT Architecture



Alberto Ballesteros

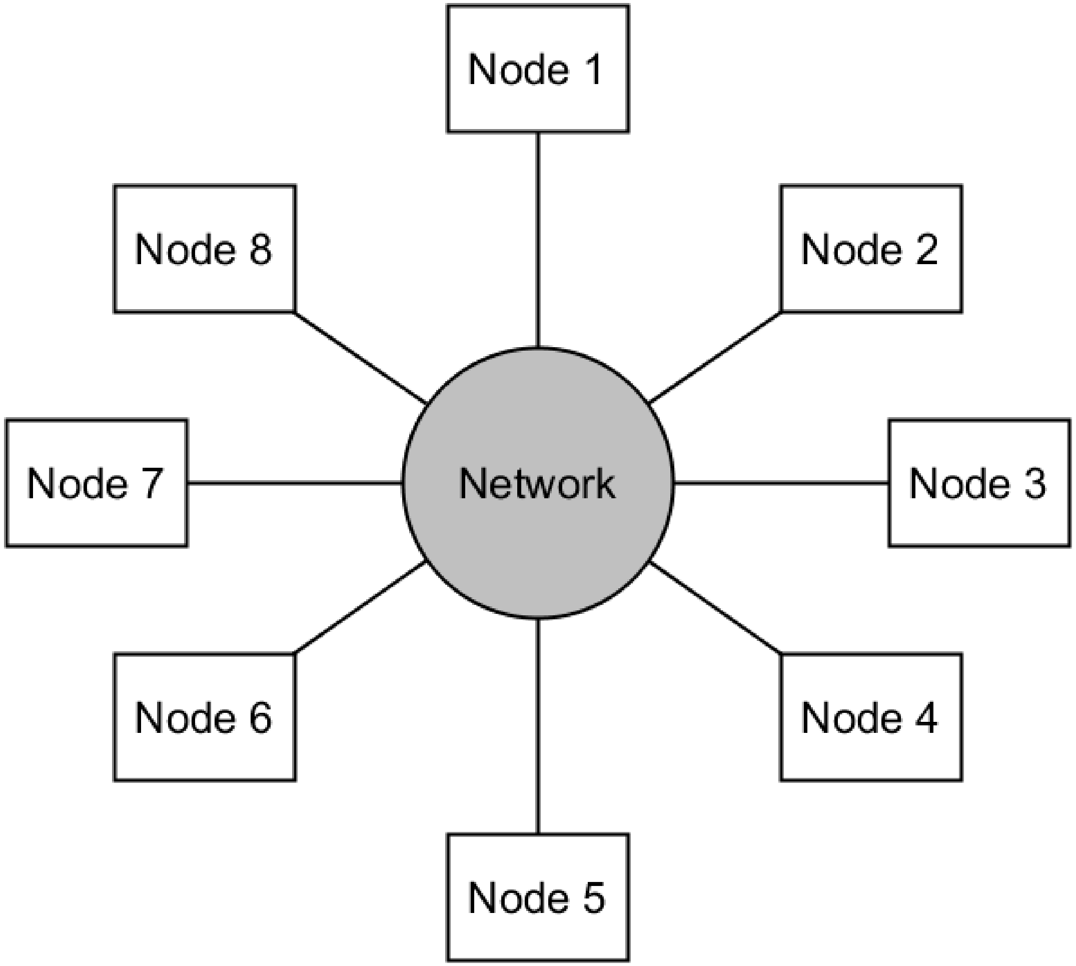
Outline

1. Motivation
2. The problem
3. The task model
4. The system architecture
5. The Knowledge Entity
6. The Wisdom Entity

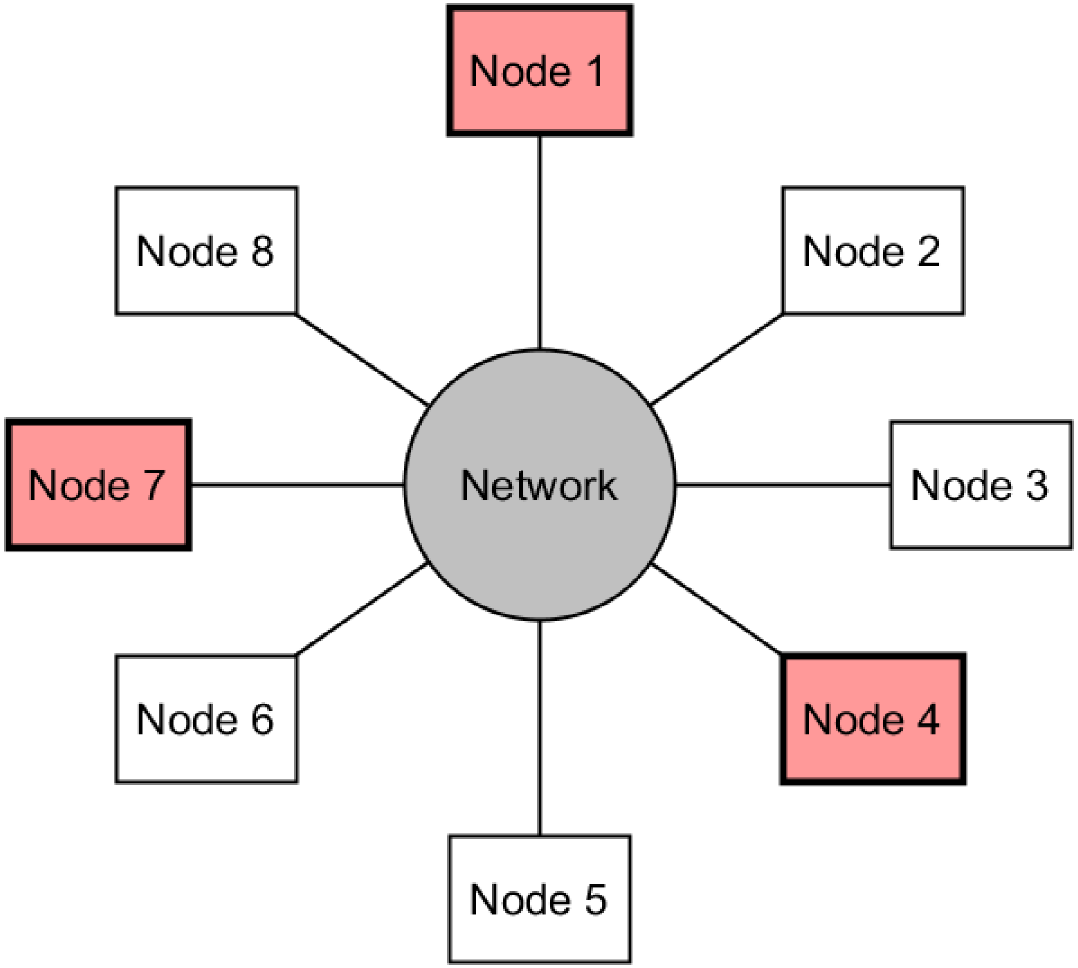
Outline

1. **Motivation**
2. The problem
3. The task model
4. The system architecture
5. The Knowledge Entity
6. The Wisdom Entity

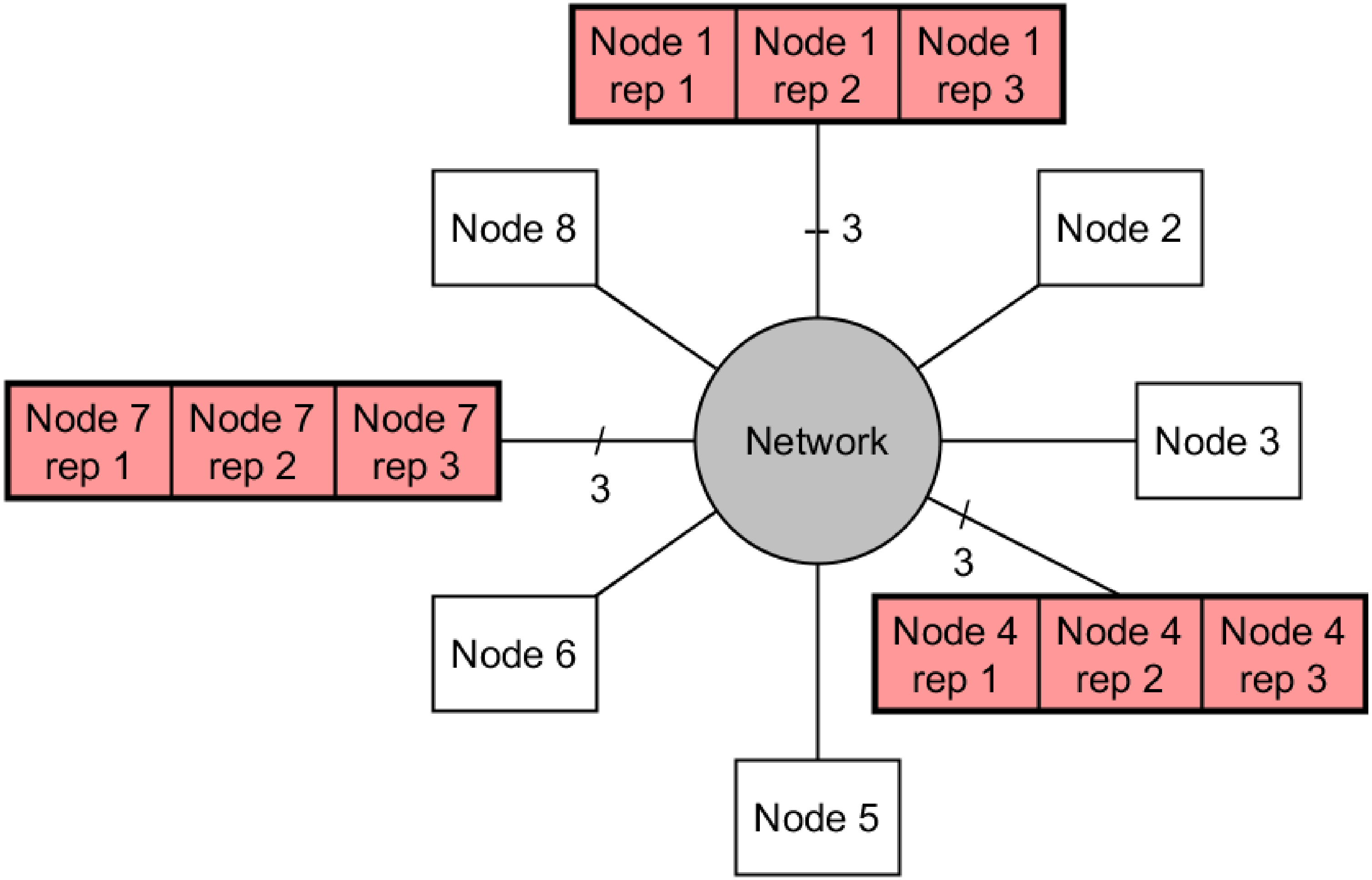
Motivation



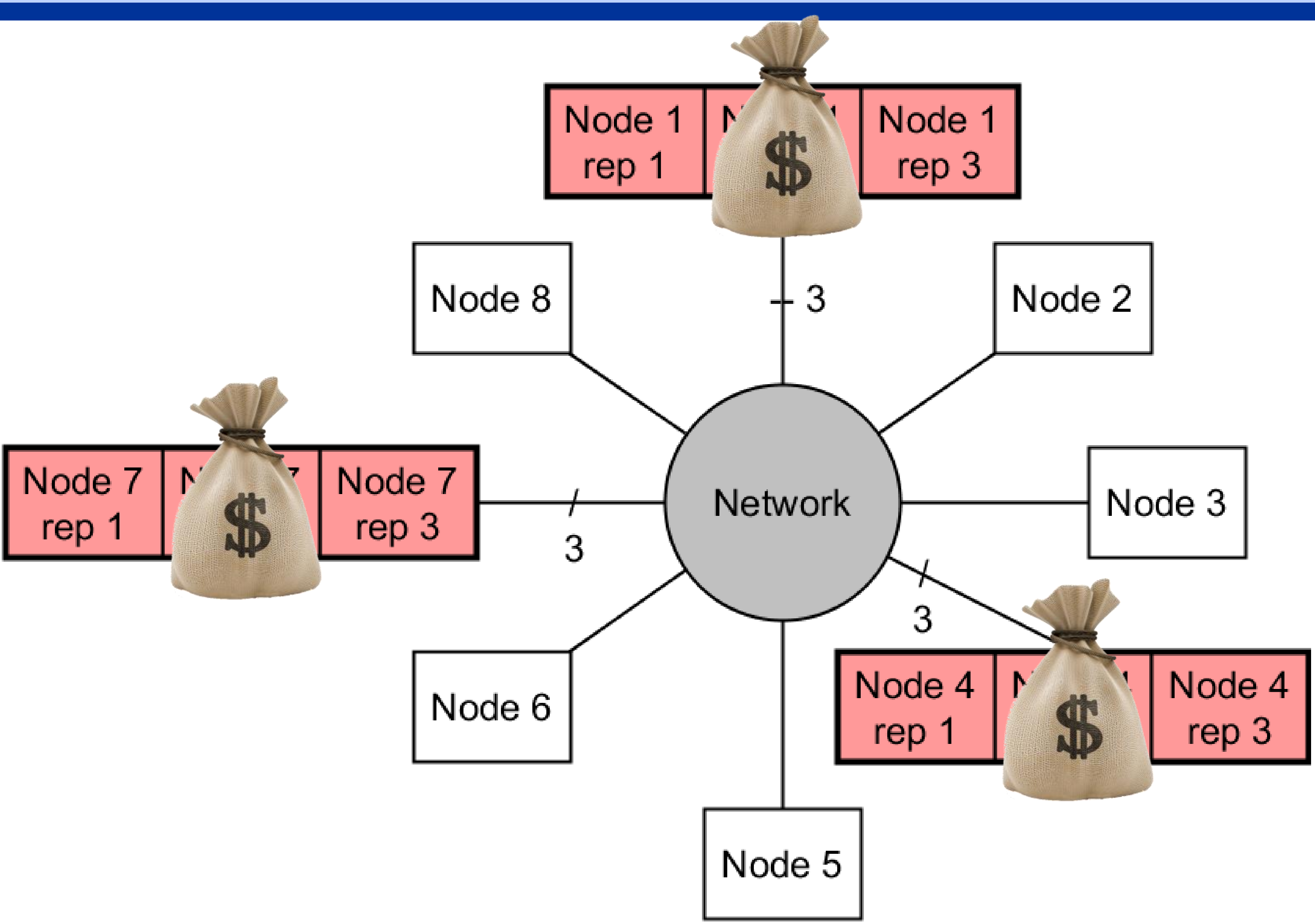
Motivation



Motivation

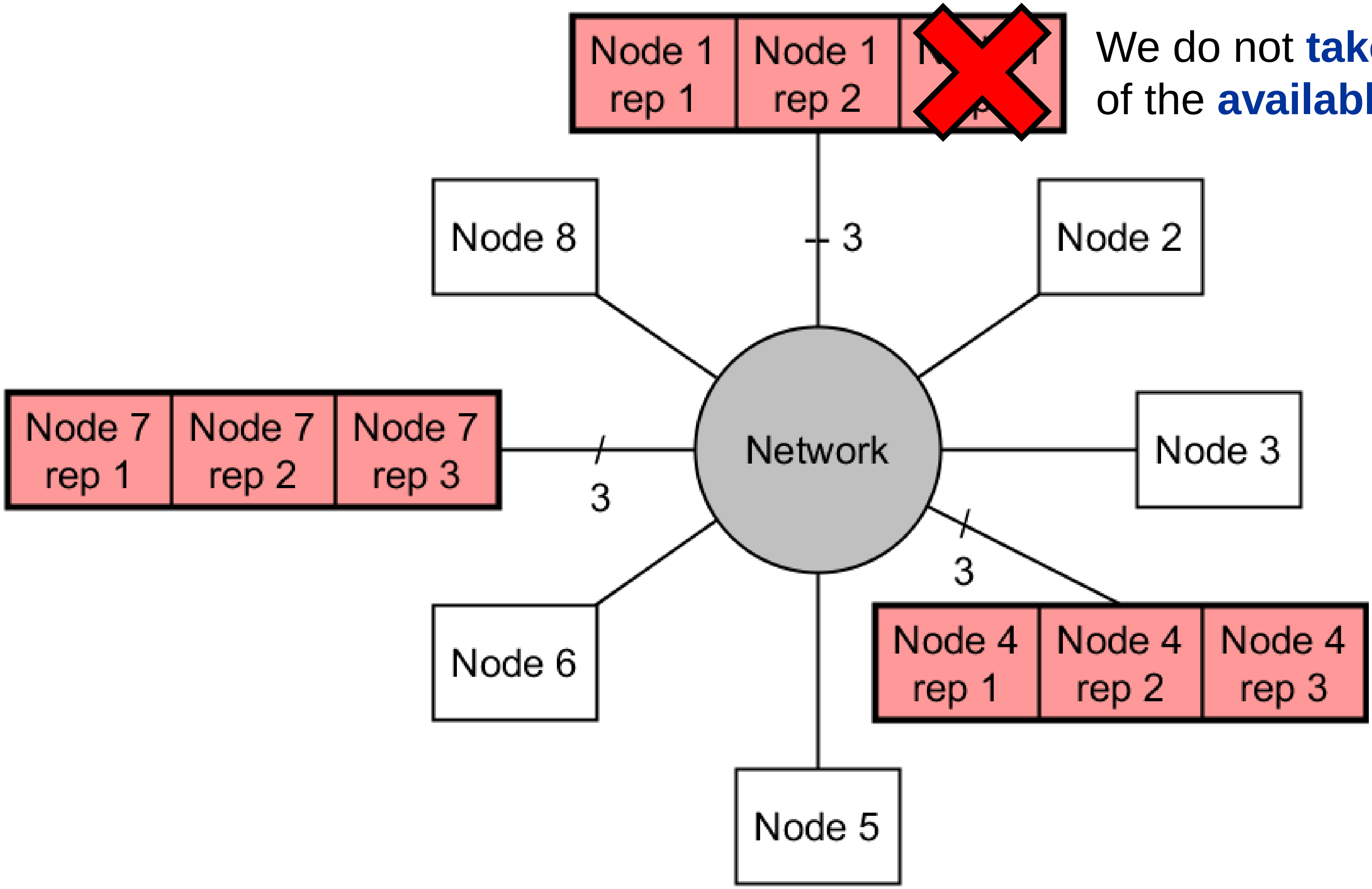

Node 1 rep 1
Node 1 rep 2
Node 1 rep 3
3
Node 8
Node 2
Node 7 rep 1
Node 7 rep 2
Node 7 rep 3
3
Network
Node 3
3
Node 6
Node 4 rep 1
Node 4 rep 2
Node 4 rep 3
Node 5

Motivation



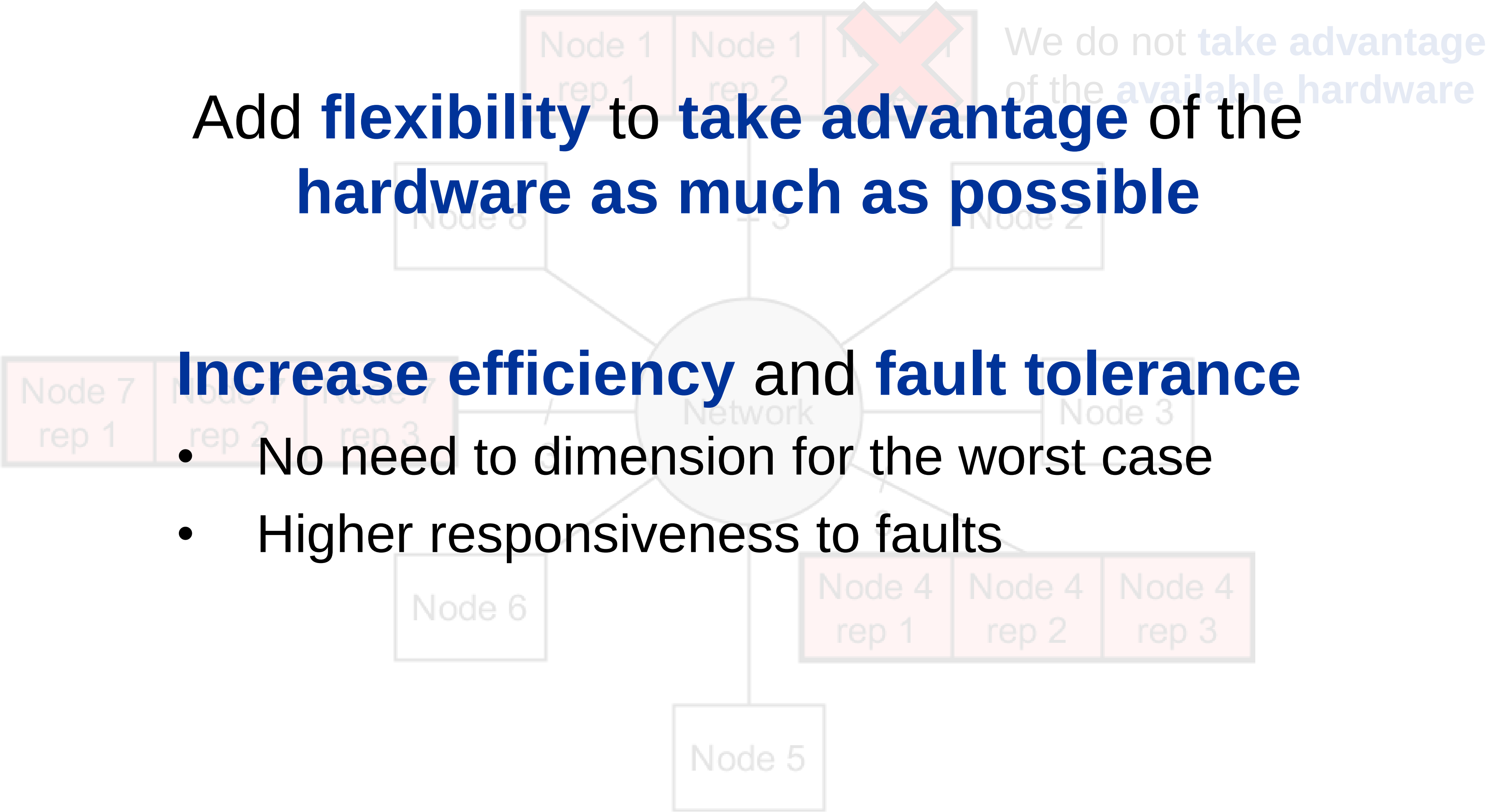
Node 1 rep 1
N 1
Node 1 rep 3
Node 8
3
Node 2
Node 7 rep 1
N 7
Node 7 rep 3
3
Network
Node 3
3
Node 6
Node 4 rep 1
N 4
Node 4 rep 3
Node 5

Motivation



We do not **take advantage** of the **available hardware**

Motivation



Add **flexibility** to **take advantage** of the **hardware as much as possible**

Increase efficiency and **fault tolerance**

- No need to dimension for the worst case
- Higher responsiveness to faults

Outline

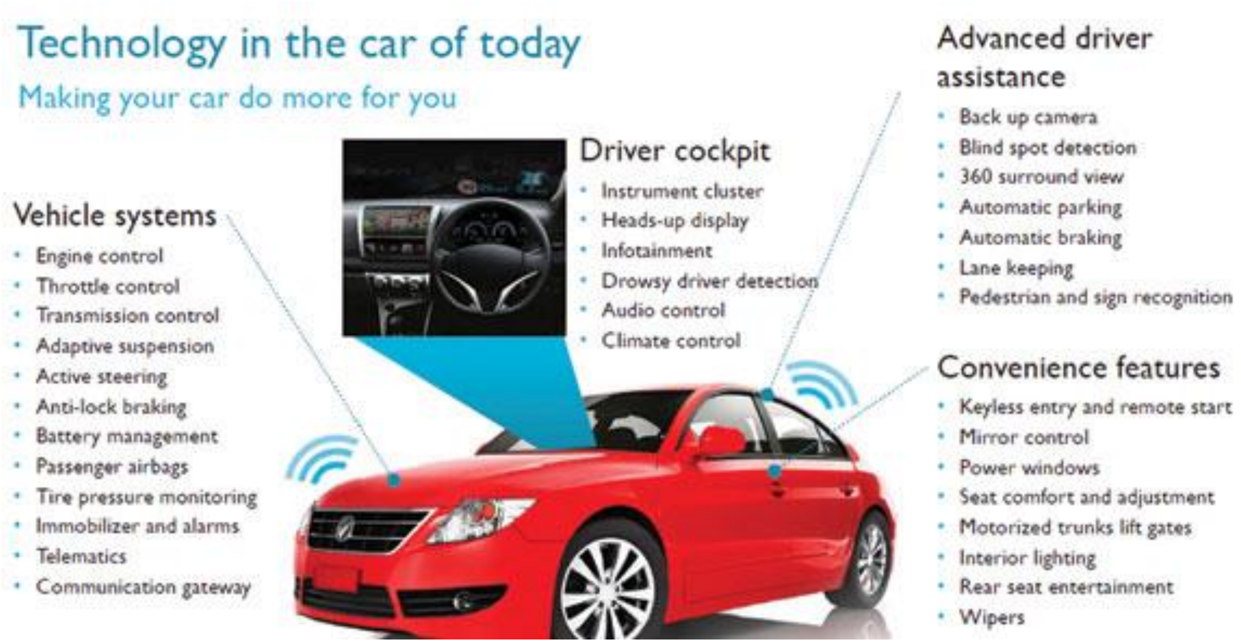
1. Motivation
2. **The problem**
3. The task model
4. The system architecture
5. The Knowledge Entity
6. The Wisdom Entity

The problem

Func 1 Func 2 Func 3 Func 4 Func 5 Func 6 ⋯ Func M

The problem

Func 1 | Func 2 | Func 3 | Func 4 | Func 5 | Func 6 ··· Func M



Technology in the car of today

Making your car do more for you

Vehicle systems

- Engine control
- Throttle control
- Transmission control
- Adaptive suspension
- Active steering
- Anti-lock braking
- Battery management
- Passenger airbags
- Tire pressure monitoring
- Immobilizer and alarms
- Telematics
- Communication gateway

Driver cockpit

- Instrument cluster
- Heads-up display
- Infotainment
- Drowsy driver detection
- Audio control
- Climate control

Advanced driver assistance

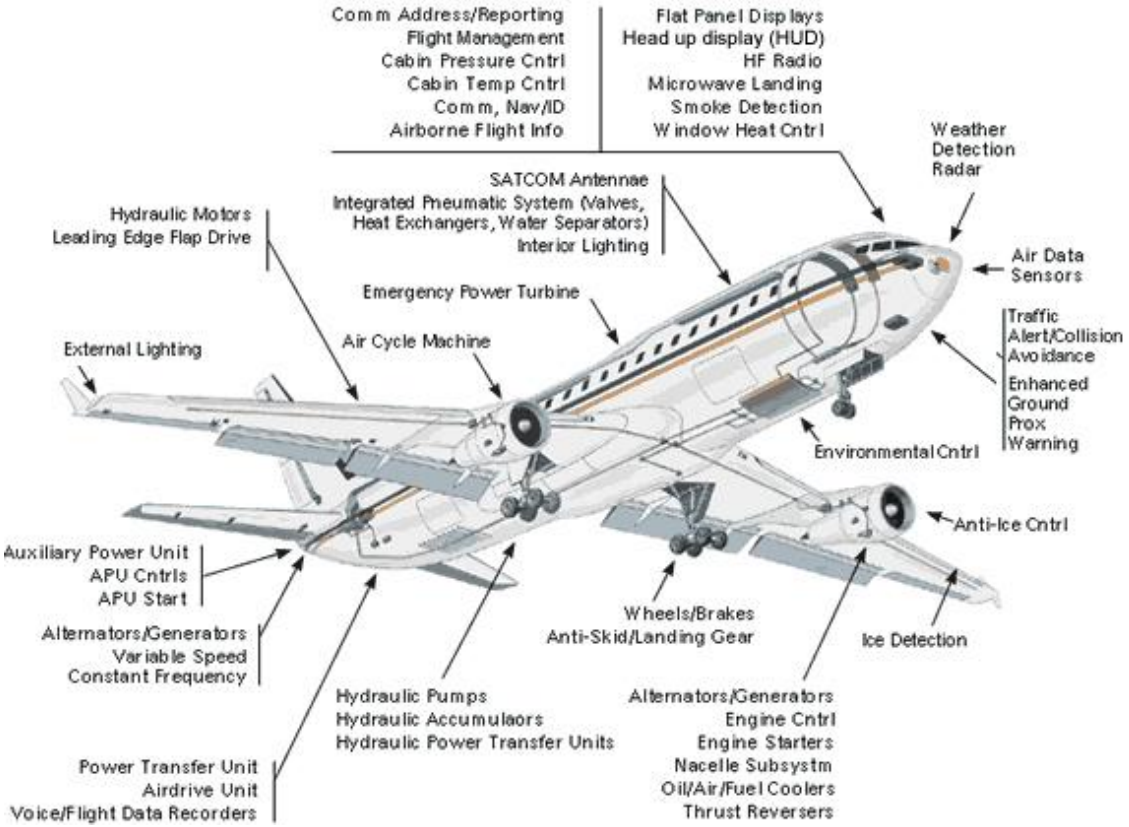
- Back up camera
- Blind spot detection
- 360 surround view
- Automatic parking
- Automatic braking
- Lane keeping
- Pedestrian and sign recognition

Convenience features

- Keyless entry and remote start
- Mirror control
- Power windows
- Seat comfort and adjustment
- Motorized trunks lift gates
- Interior lighting
- Rear seat entertainment
- Wipers

The problem

Func 1 | Func 2 | Func 3 | Func 4 | Func 5 | Func 6 | ··· | Func M



The problem

Func 1 Func 2 Func 3 Func 4 Func 5 Func 6 ⋯ Func M

Correct operation of the system

- Execute the indispensable functionalities
- Fulfil the operational requirements of the functionalities

The problem

Func 1 Func 2 Func 3 Func 4 Func 5 Func 6 ··· Func M

Correct operation of the system

- Execute the indispensable functionalities
- **Fulfil the operational requirements of the functionalities**

The problem

Func 1 Func 2 Func 3 Func 4 Func 5 Func 6 ··· Func M

Func 1: Multimedia

Func 2: Control

Func 3: Logging

The problem

Func 1
Func 2
Func 3
Func 4
Func 5
Func 6
···
Func M

Func 1: Multimedia (**NRT**, **NRG**[*1])

Func 2: Control (**HRT**, **HRG**[*1])

Func 3: Logging (**SRT**, **HRG**[*1])

[*1] RG: Reliability Goal

The problem

Func 1: Multimedia (**NRT**, **NRG**[*1])

Func 2: Control (**HRT**, **HRG**[*1])

Func 3: Logging (**SRT**, **HRG**[*1])

Real Time requirements

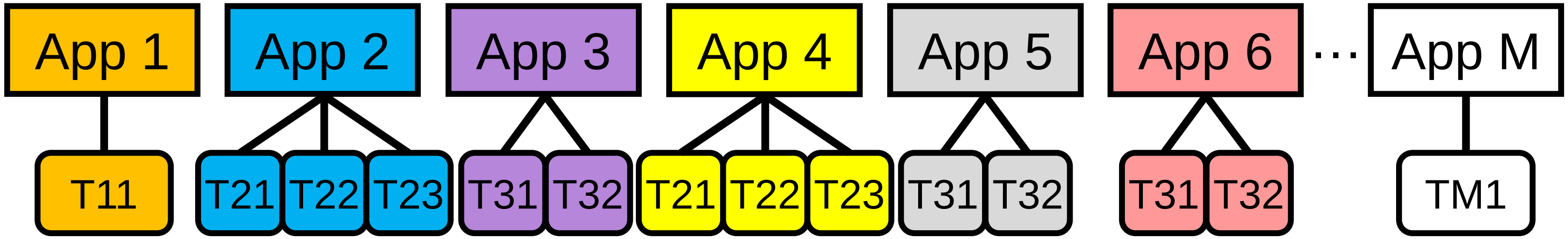
- NRT, SRT and HRT
- Desired T_{app} and D_{app}
- Minimum T_{app} and D_{app}

Reliability requirements

- NRG and HRG
- Number of 9s

[*1] RG: Reliability Goal

The problem



Func 1: Multimedia (**NRT**, **NRG**[*1])

Func 2: Control (**HRT**, **HRG**[*1])

Func 3: Logging (**SRT**, **HRG**[*1])

Real Time attributes

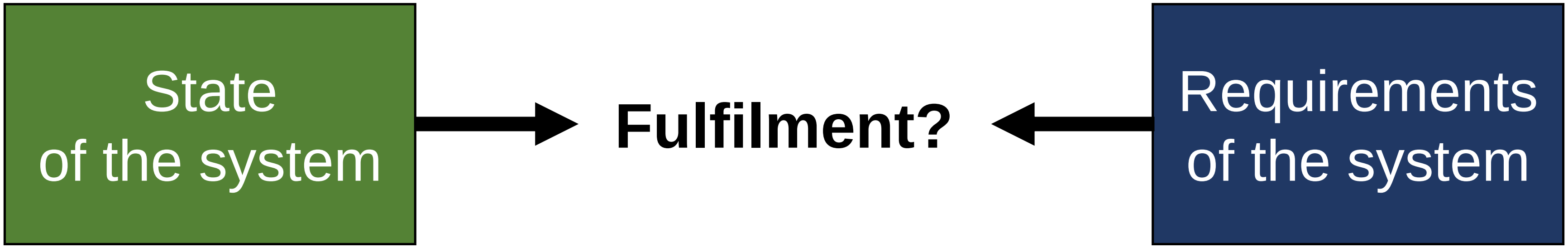
- T_{task} and D_{task}

Fault tolerance attributtes

- Number of replicas

[*1] RG: Reliability Goal

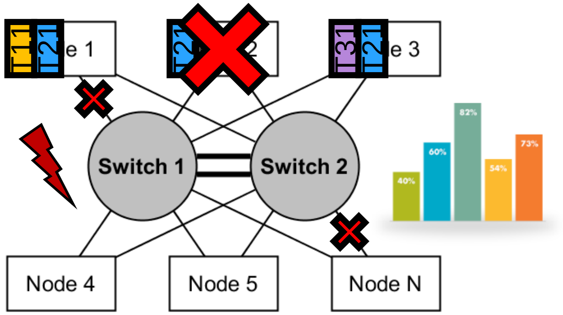
The problem



State of the system

Fulfilment?

Requirements of the system



List of tasks

RT requirements
R(t) requirements

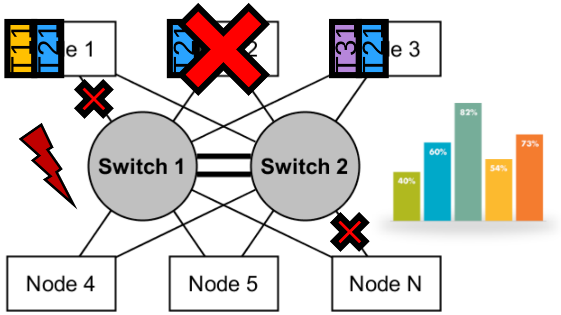
The problem

State
of the system

Fulfilment?

Requirements
of the system

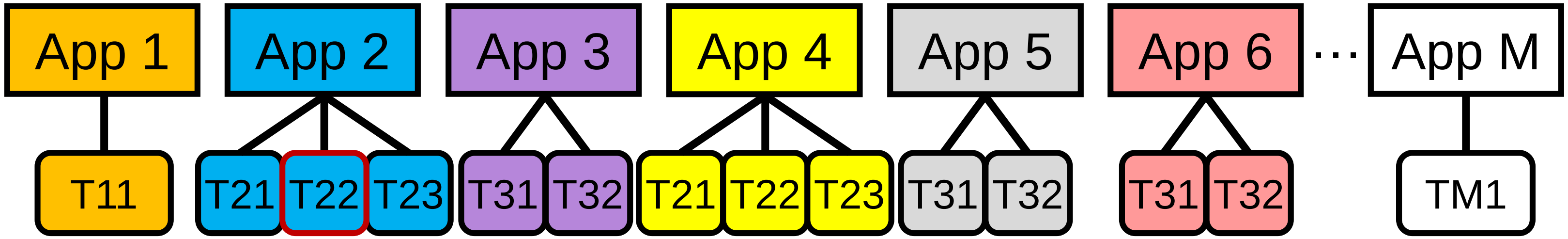
Reconfigure



List of tasks

RT requirements
R(t) requirements

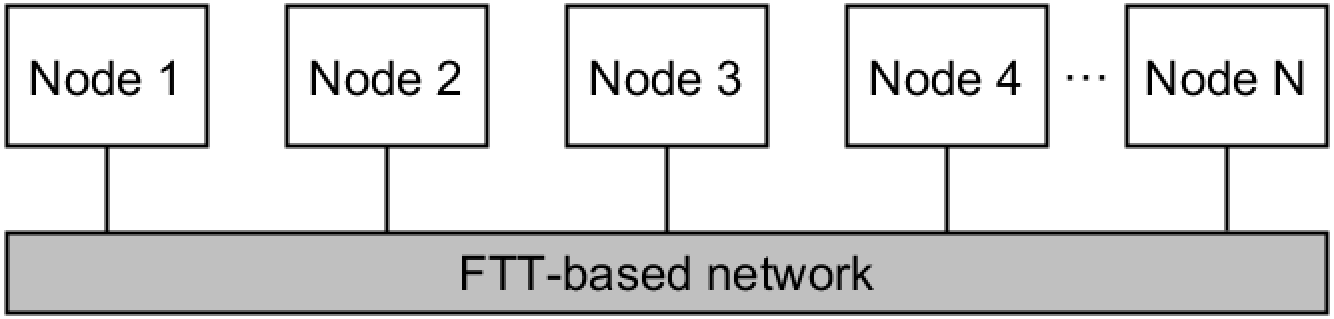
The problem



App 1 · App 2 · App 3 · App 4 · App 5 · App 6 · ··· · App M

T11 · T21 T22 T23 · T31 T32 · T21 T22 T23 · T31 T32 · T31 T32 · TM1

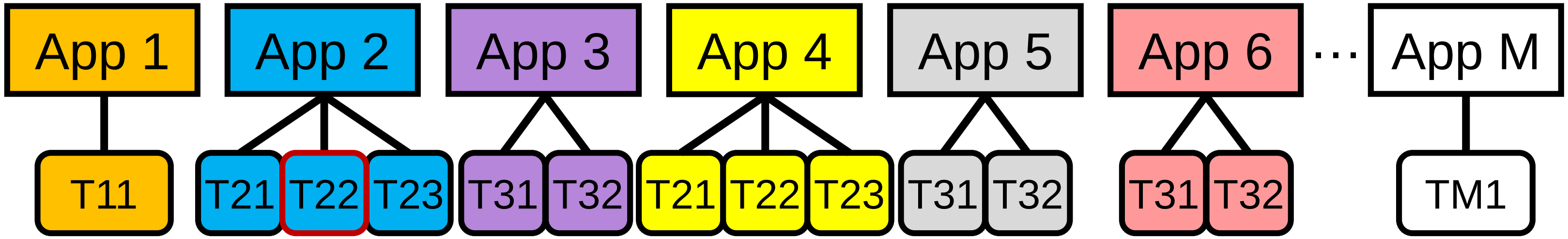
- start/stop app
- start/stop task
- modify real-time and comm. attributes



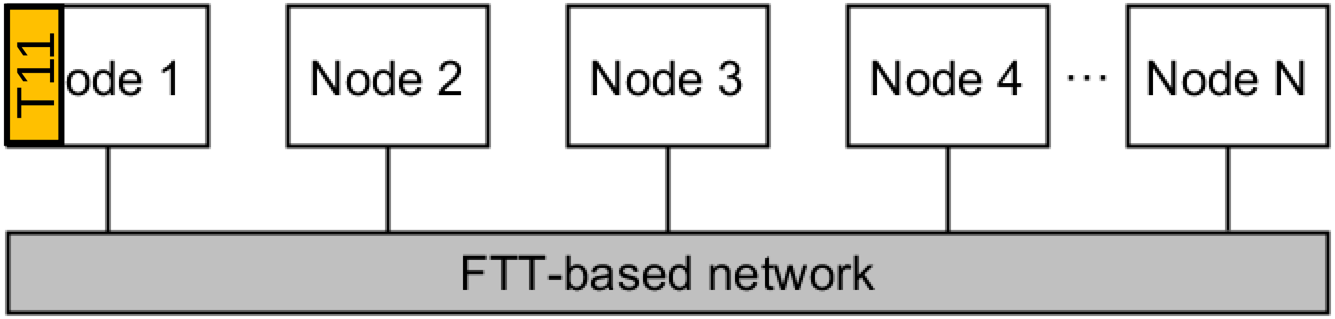
Node 1 · Node 2 · Node 3 · Node 4 · ··· · Node N

FTT-based network

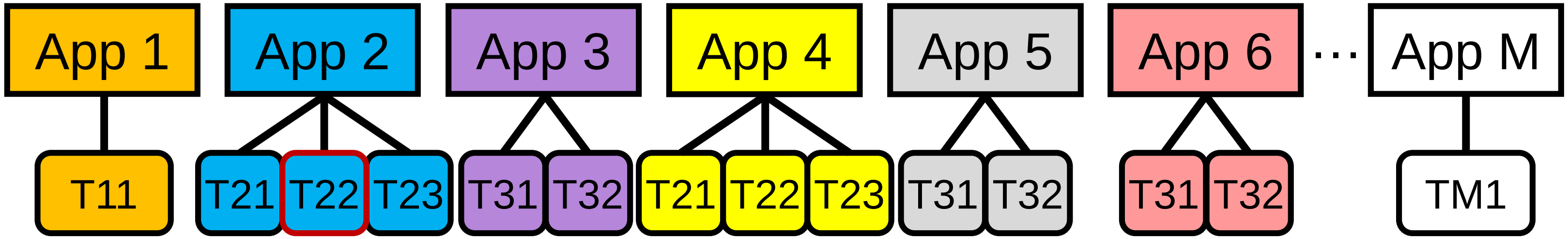
The problem



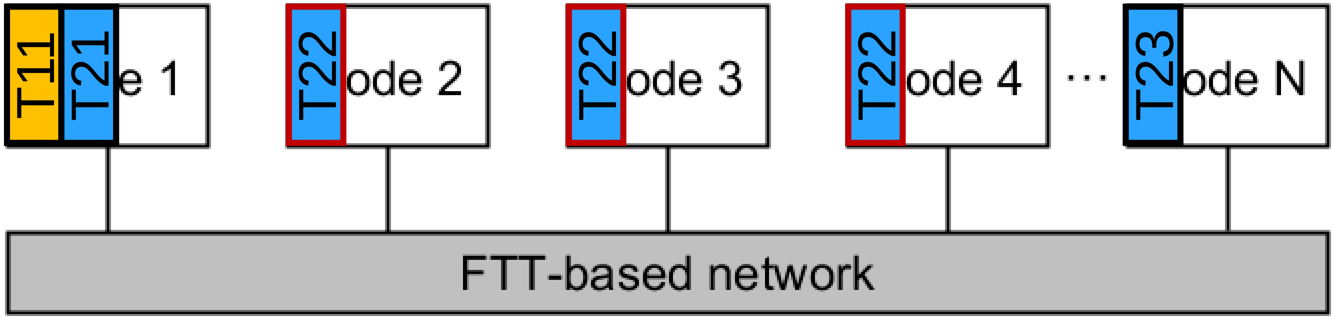
- **start**/stop **app**
- start/stop task
- modify real-time and comm. attributes



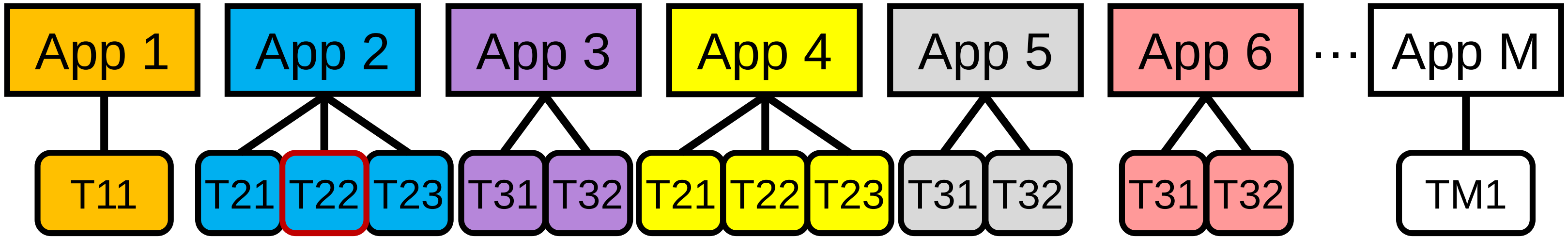
The problem



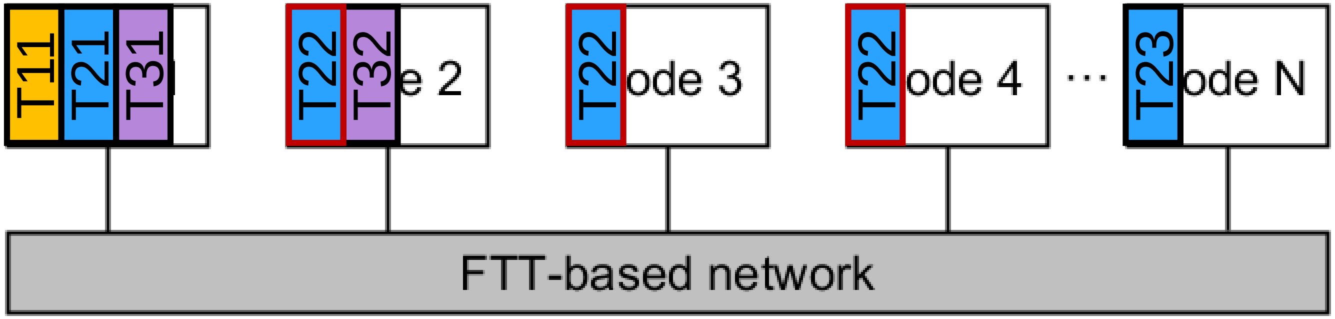
- **start**/stop **app**
- start/stop task
- modify real-time and comm. attributes



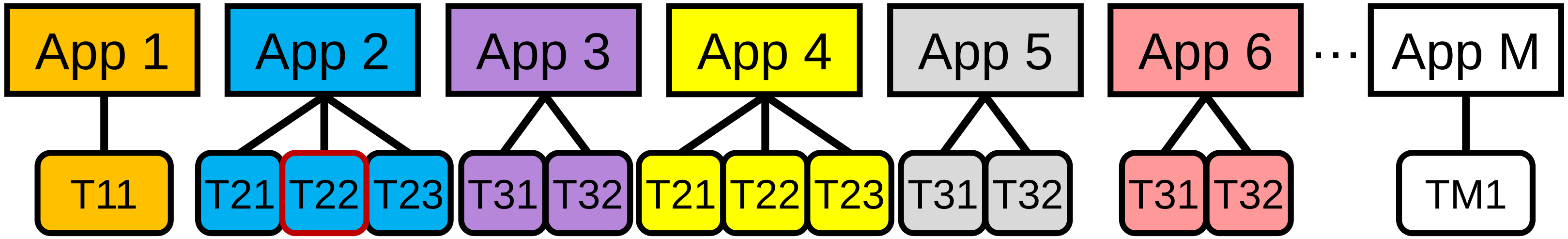
The problem



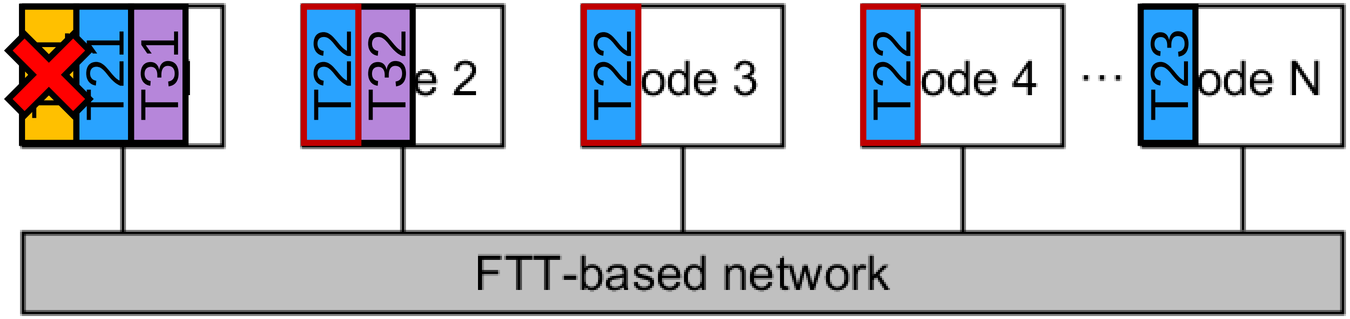
- **start**/stop **app**
- start/stop task
- modify real-time and comm. attributes



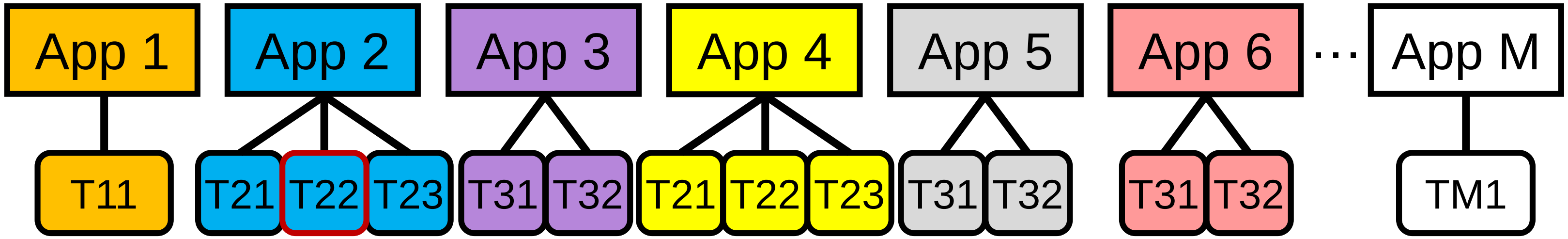
The problem



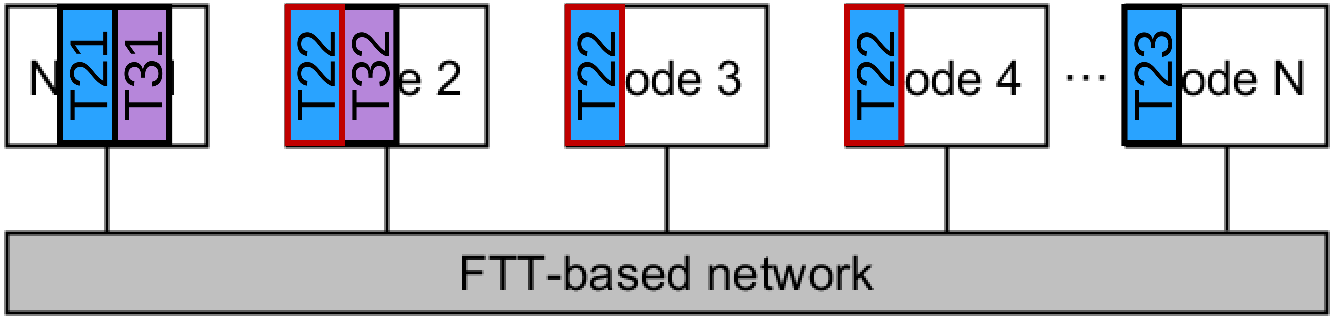
- start/**stop app**
- start/stop task
- modify real-time and comm. attributes



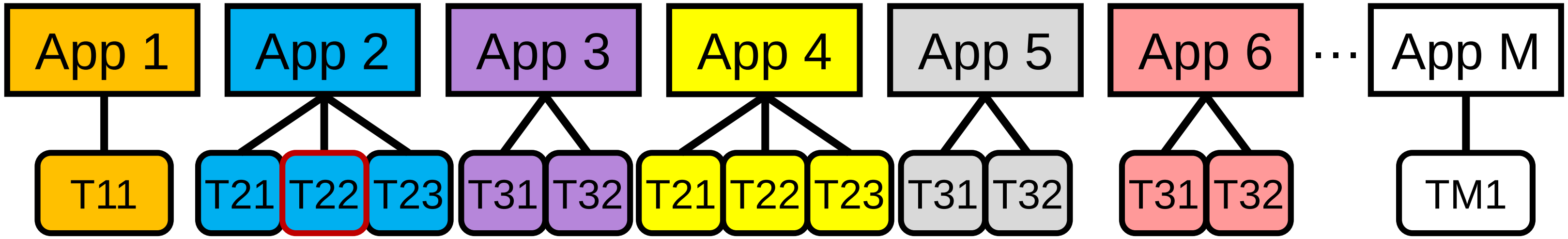
The problem



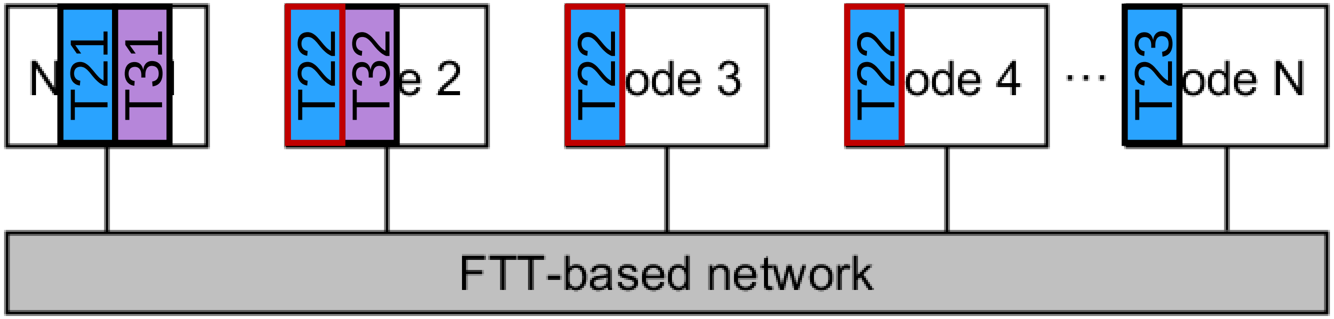
- start/**stop app**
- start/stop task
- modify real-time and comm. attributes



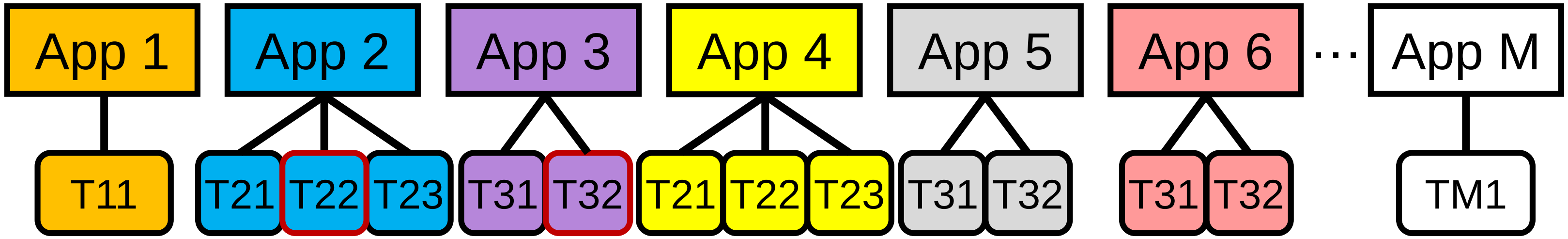
The problem



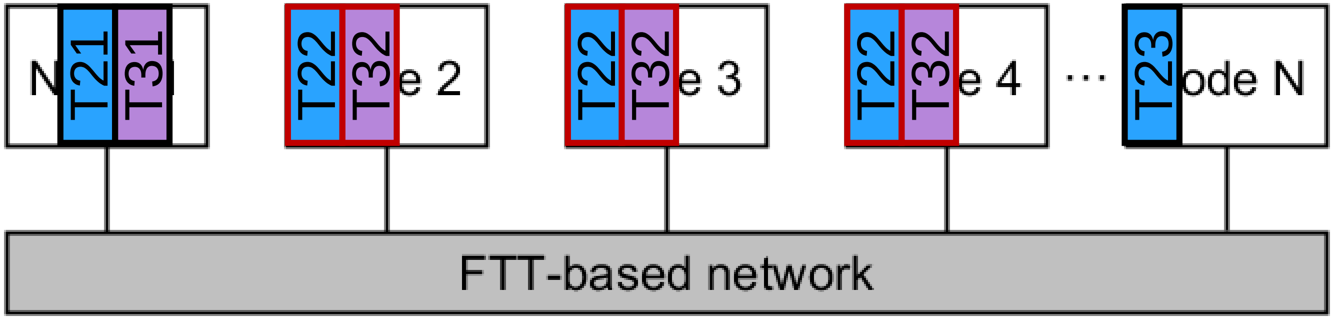
- start/stop app
- **start**/stop **task** (change level of repl. and reallocate)
- modify real-time and comm. attributes



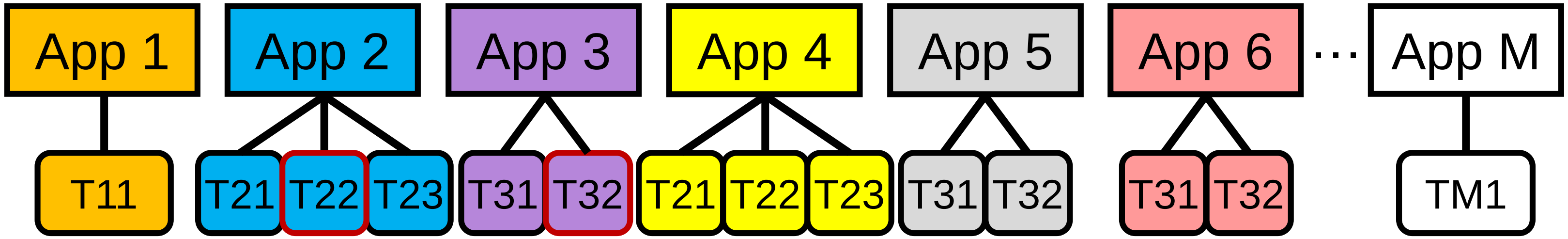
The problem



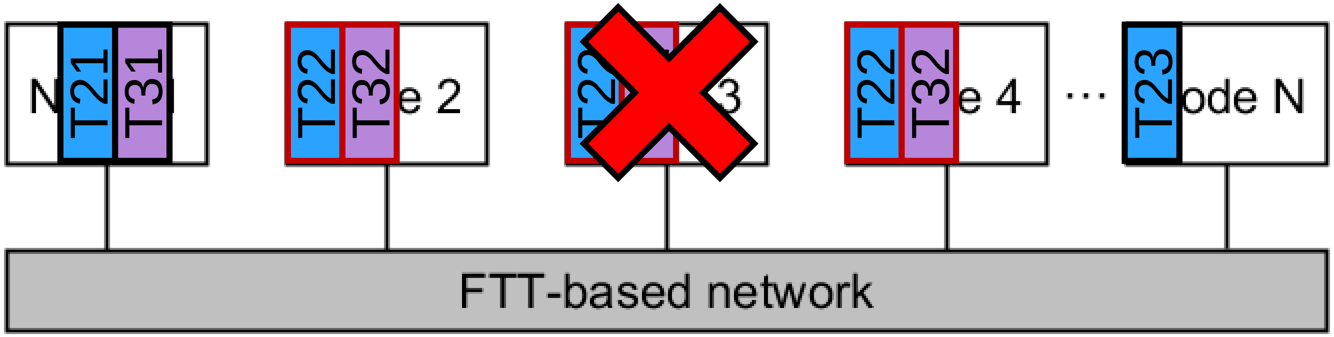
- start/stop app
- **start**/stop **task** (change level of repl. and reallocate)
- modify real-time and comm. attributes



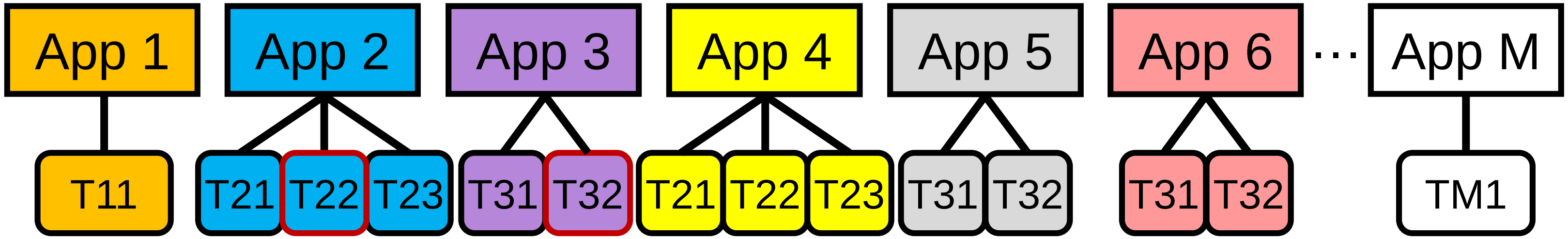
The problem



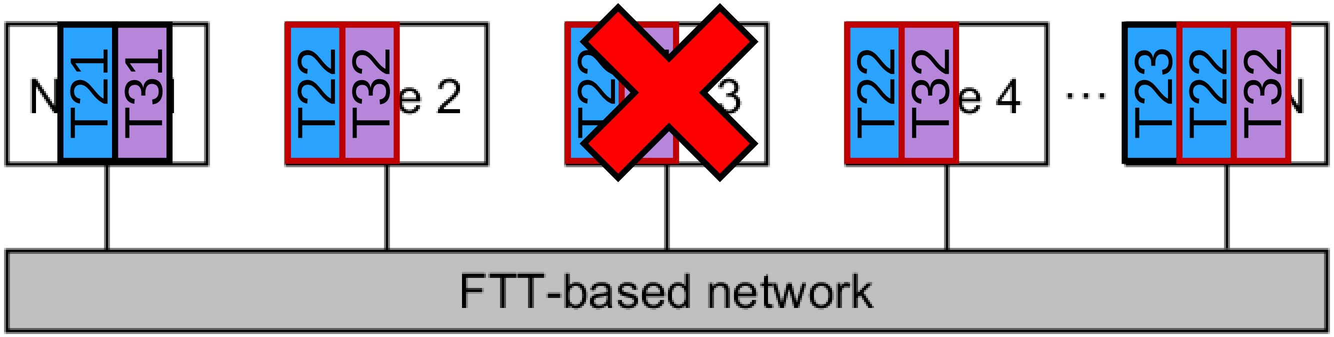
- start/stop app
- **start**/stop **task** (change level of repl. and reallocate)
- modify real-time and comm. attributes



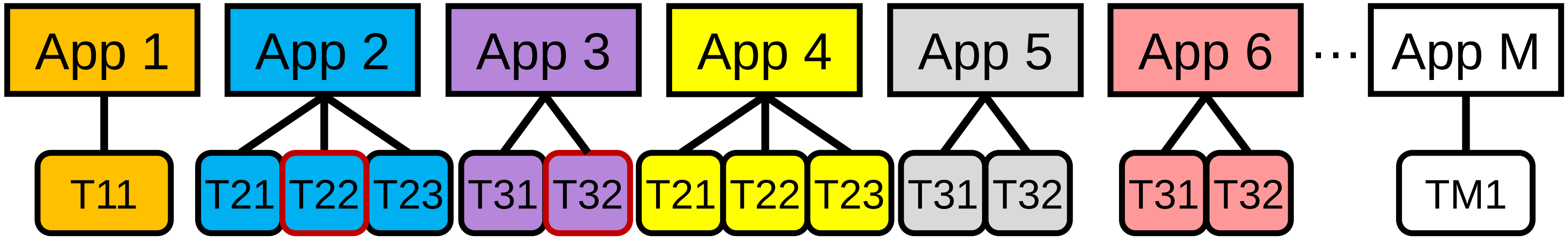
The problem



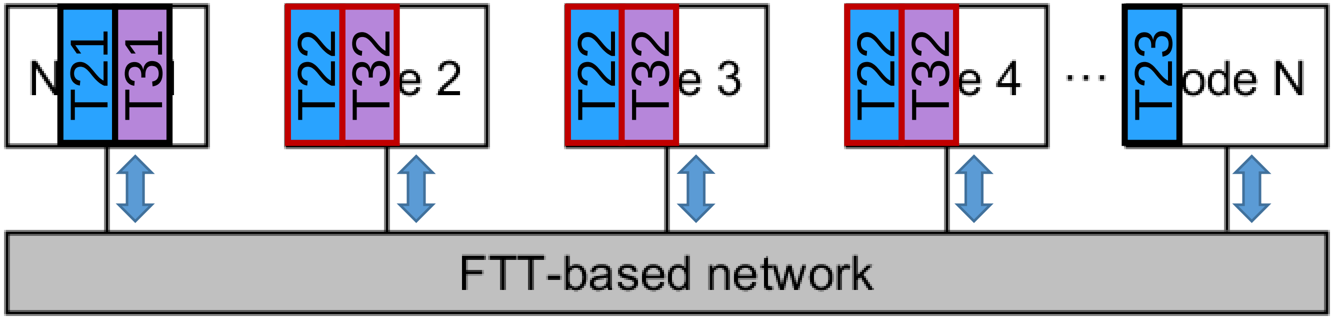
- start/stop app
- **start**/stop **task** (change level of repl. and reallocate)
- modify real-time and comm. attributes



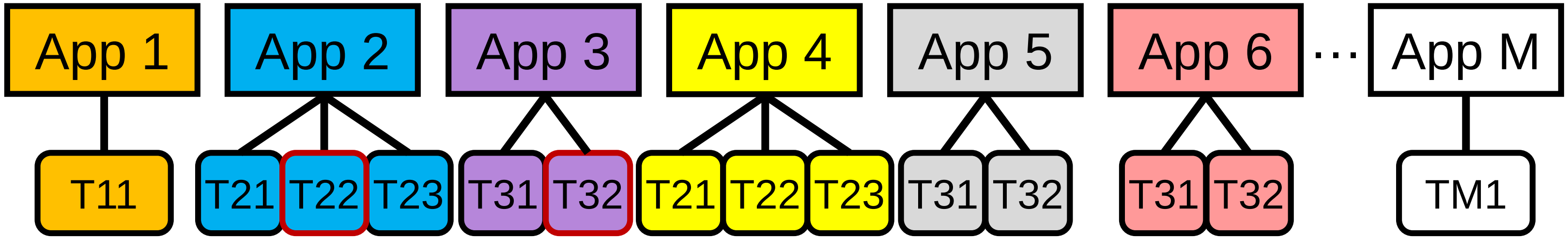
The problem



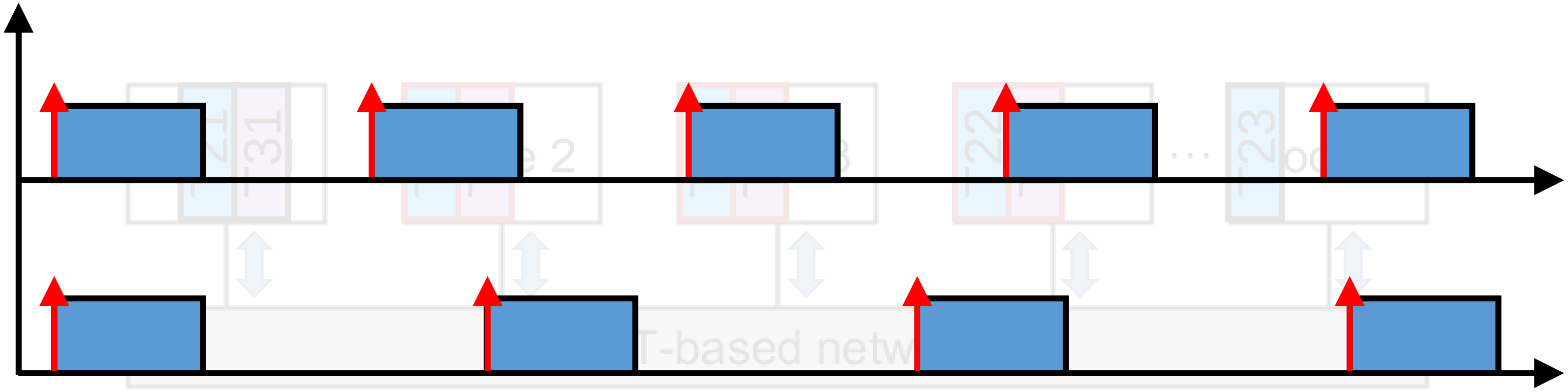
- start/stop app
- start/stop task
- **modify real-time and comm. attributes**



The problem



- start/stop app
- start/stop task
- **modify real-time and comm. attributes**



Outline

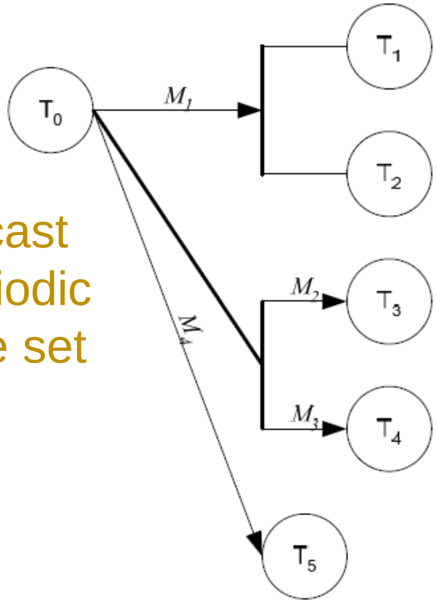
1. Motivation
2. The problem
3. **The task model**
4. The system architecture
5. The Knowledge Entity
6. The Wisdom Entity

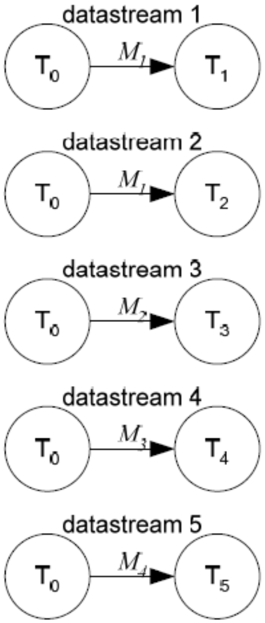
The task model

Extend the task model based on **Data Streams** proposed by Calha [1]

A **data stream** represents an **information flow** between **one producer task** to **one or more consumer tasks**

- Unicast/Multicast
- Periodic/Aperiodic
- Fixed/Variable set





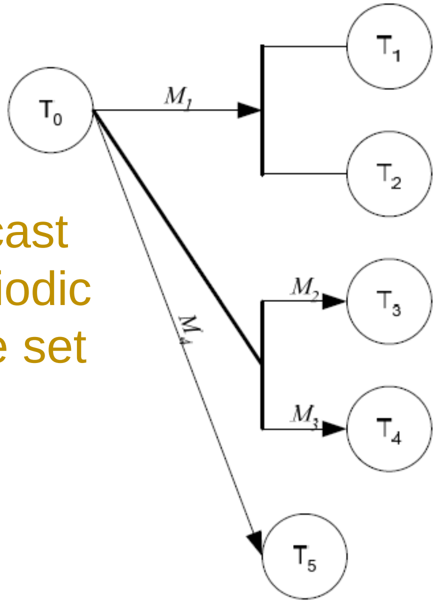
[1] A Holistic Approach Towards Flexible Distributed Systems

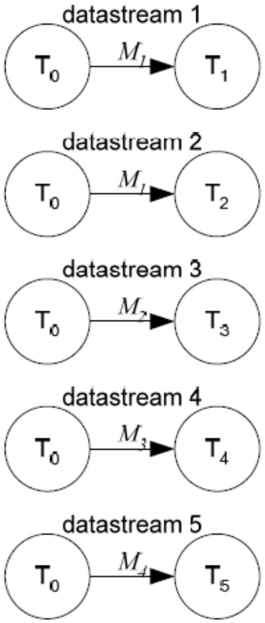
The task model

Extend the task model based on **Data Streams** proposed by Calha [1]

Consumer tasks may also produce data to other tasks thus becoming **consumer/producer tasks**

- Unicast/Multicast
- Periodic/Aperiodic
- Fixed/Variable set





[1] A Holistic Approach Towards Flexible Distributed Systems

The task model

C_i: Worst-case computation time
T_i: Period
Ph_i: Phase (relative to period) (first release instant)
D_i: Deadline (relative to release instant)
N_i: Node where the task runs
MP_i: Message produced
MC_i: Message consumed
Pr_i: Priority (in case of fixed-priority scheduling)

$d_{i,k}$: absolute deadline of instance *k*
$r_{i,k}$: release instant of instance *k*

$$T = \{ t_{i,k} \ (C_i, T_i, Ph_i, D_i, N_i, MP_i, MC_i, Pr_i, \ d_{i,k}, r_{i,k}),$$
$$i=1..NUM\_TASKS, \ k=1..NUM\_INSTS \ \}$$

The task model

$\mathbf{C_j}$: Worst-case transmission time

$\mathbf{T_j}$: Period

$\mathbf{Ph_j}$: Phase (relative to period) (first release instant)

$\mathbf{D_j}$: Deadline (relative to release instant)

$\mathbf{PT_j}$: Producer task

$\mathbf{CTL_{j,i}}$: Consumer task list

$\mathbf{Pr_j}$: Priority (in case of fixed-priority scheduling)

$\mathbf{d_{i,k}}$: absolute deadline of instance *k*

$\mathbf{r_{i,k}}$: release instant of instance *k*

$$S = \{\ s_{j,k}\ (C_j, T_j, Ph_j, D_j, PT_j, CTL_{j,i},\ d_{j,k}, r_{j,k}),$$
$$j=1..\text{NUM\_STREAMS},\ i=1..\text{NUM\_TASKS},\ k=1..\text{NUM\_INSTS}\ \}$$

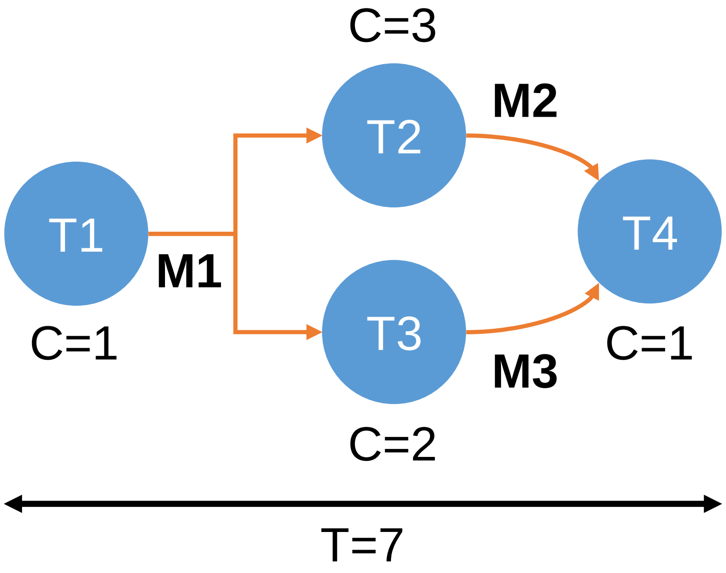
The task model

The purpose of this work is to **determine and tune** the **parameters** related to the **triggering of tasks and messages**
→ phases, periods and deadlines
in a **centralized, online** and **holistic manner**

The **parameter determination** is based on the **most constrained resource**, the network or the nodes, and the **relation between the execution and transmission windows** of related tasks and messages

- Node-centric
  - Low network load and high node load
  - Tasks impose restrictions to the set of messages

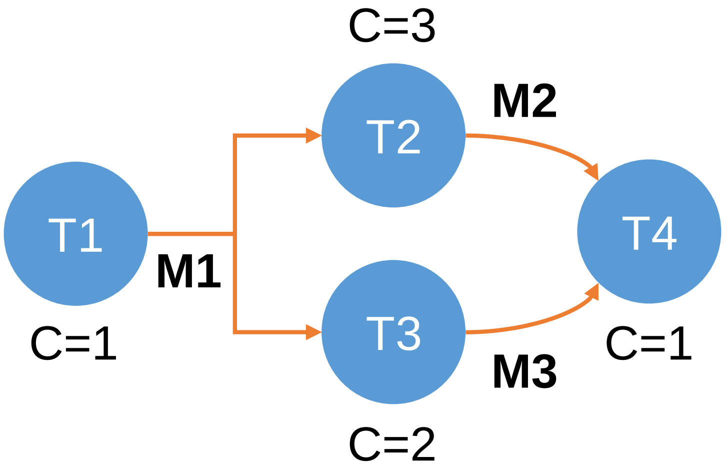
The task model



Task	C	T	D	MC	MP
T1	1	7	1	-	M1
T2	3	7	3	M1	M2
T3	2	7	2	M1	M3
T4	1	7	1	M2, M3	-

Msg	C	PT	CTL
M1	1	T1	T2, T3
M2	1	T2	T4
M3	1	T3	T4

The task model



Task				Msg		
	Ph			T	D	Ph
T1	0		M1	7	1	1
T2	2		M2	7	1	5
T3	3		M3	7	1	5
T4	6					

EC	00	01	02	03	04	05	06	07	08	09	10	11	12	13
T1	🟩							🟩						
T2			🟩	🟩	🟩					🟩	🟩	🟩		
T3				🟩	🟩						🟩	🟩		
T4							🟩							🟩
M1		🟨							🟨					
M2						🟨							🟨	
M3						🟨							🟨	

The task model

Comments

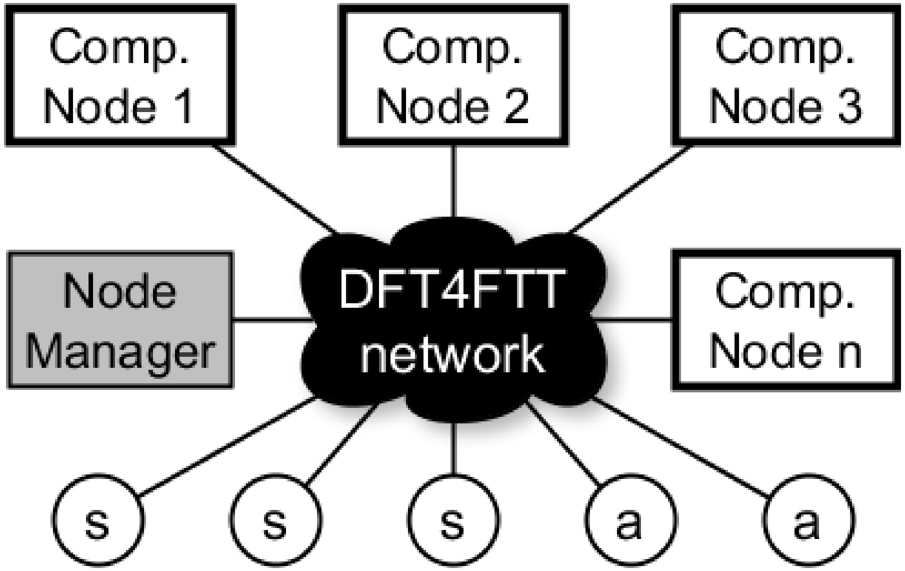
- Need to introduce **scheduling** at the **node** and **message level**
- Need to introduce the **reliability requirements**
- It **does not minimize** the number of **reconfigurations**
- **Task replication** can be implemented easily
    - Replicated streams are not needed

Outline

1. Motivation
2. The problem
3. The task model
4. **The system architecture**
5. The Knowledge Entity
6. The Wisdom Entity

The system architecture

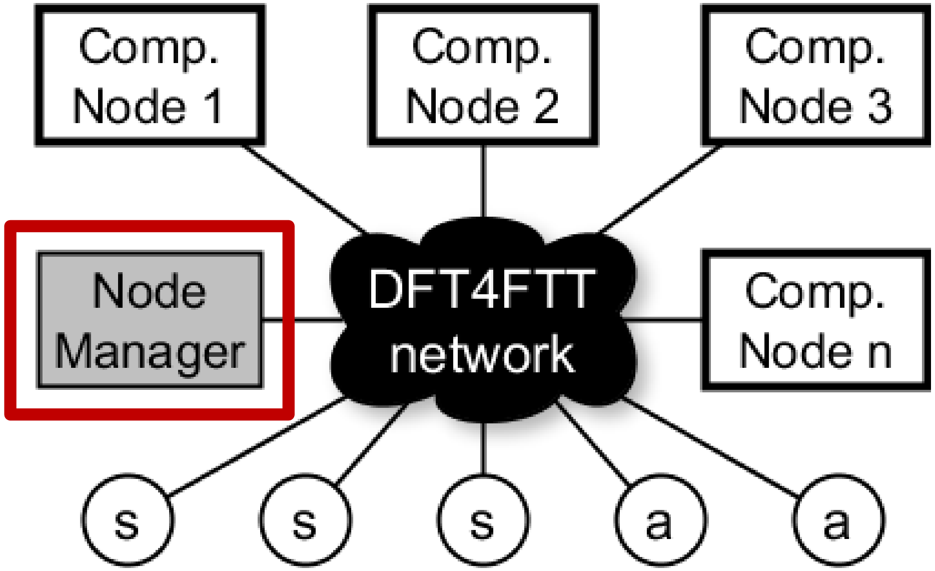
At the **node level**, the DFT4FTT architecture is composed of **various components**



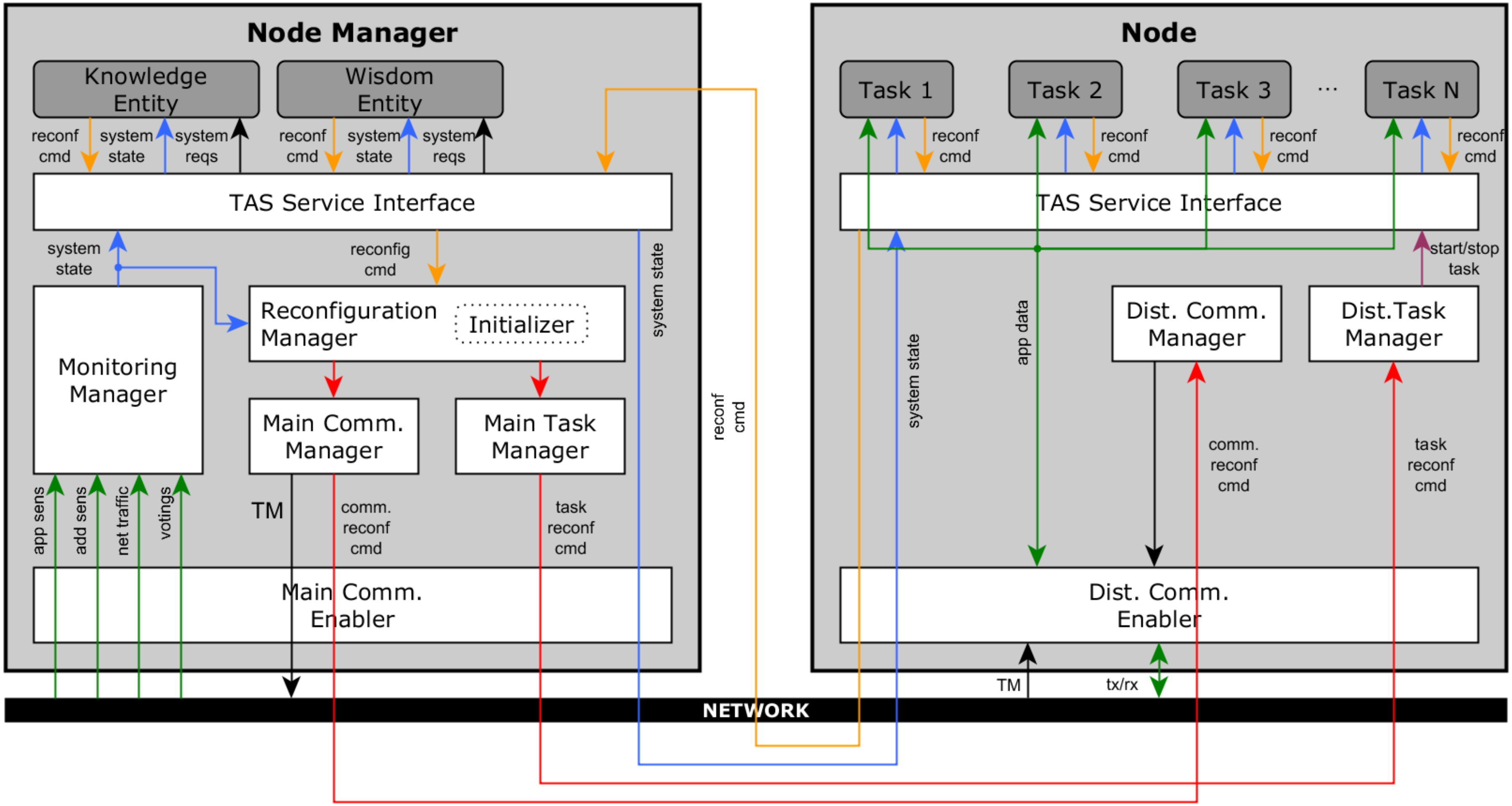
The system architecture

At the **node level**, the DFT4FTT architecture is composed of **various components**

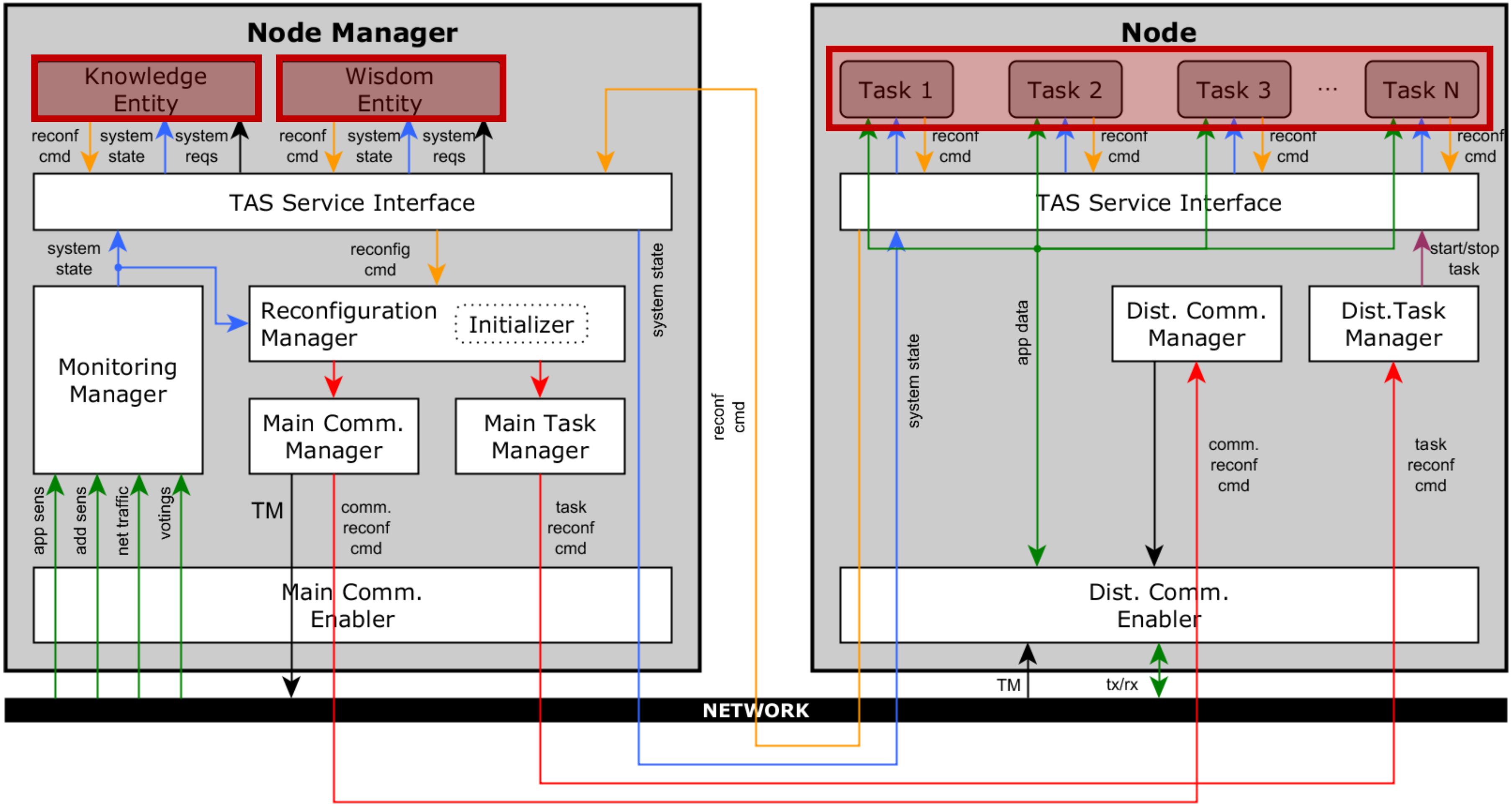
- **Monitor**
- **Detect**
- **Reconfigure**



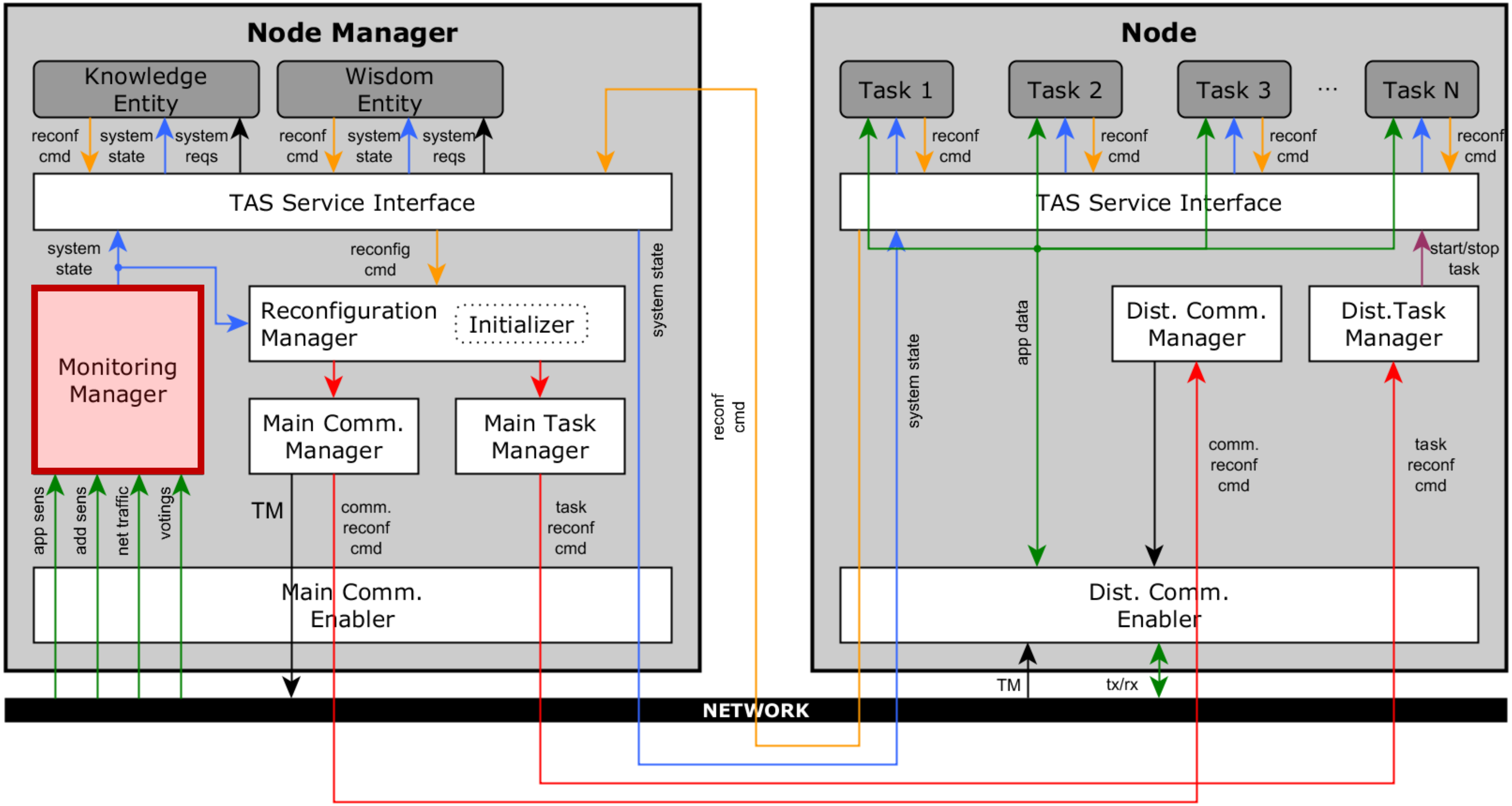
The system architecture



The system architecture



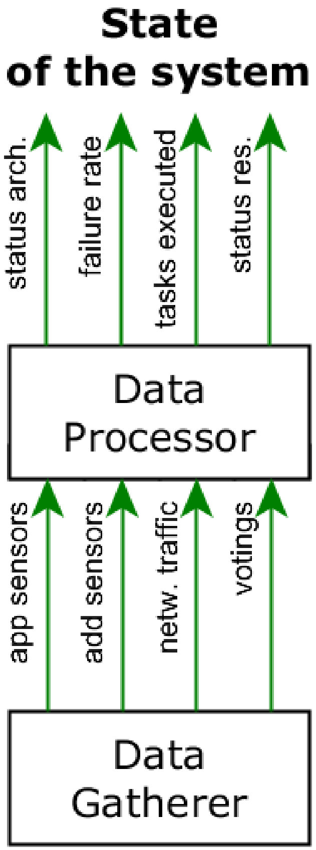
The system architecture



The system architecture

Monitoring Manager

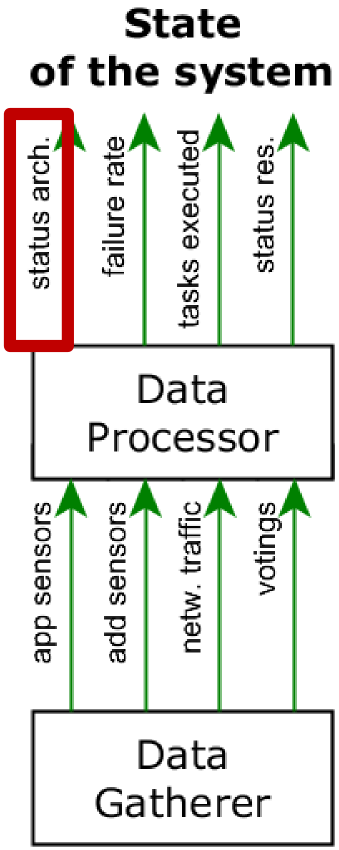
Determine and populate the **state of the system**

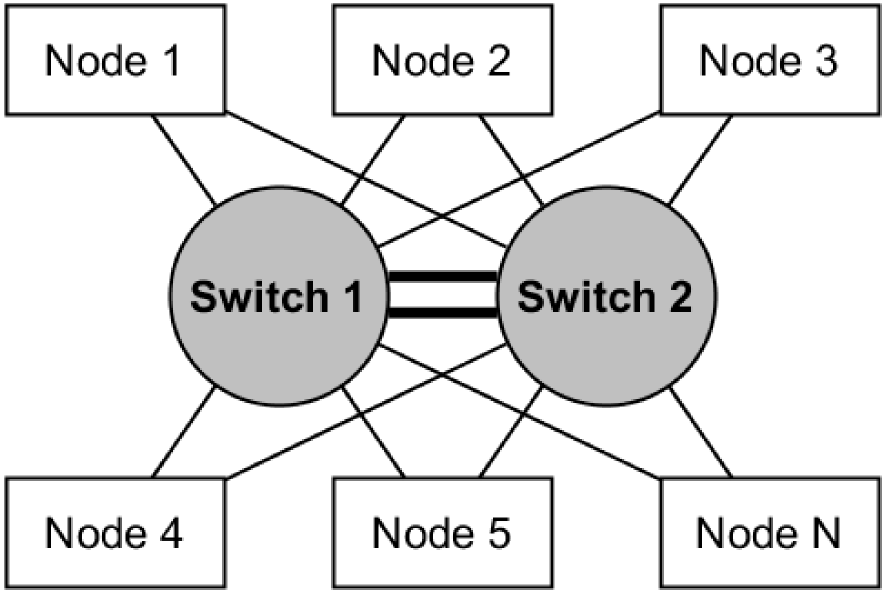


The system architecture

Monitoring Manager

Determine and populate the **state of the system**

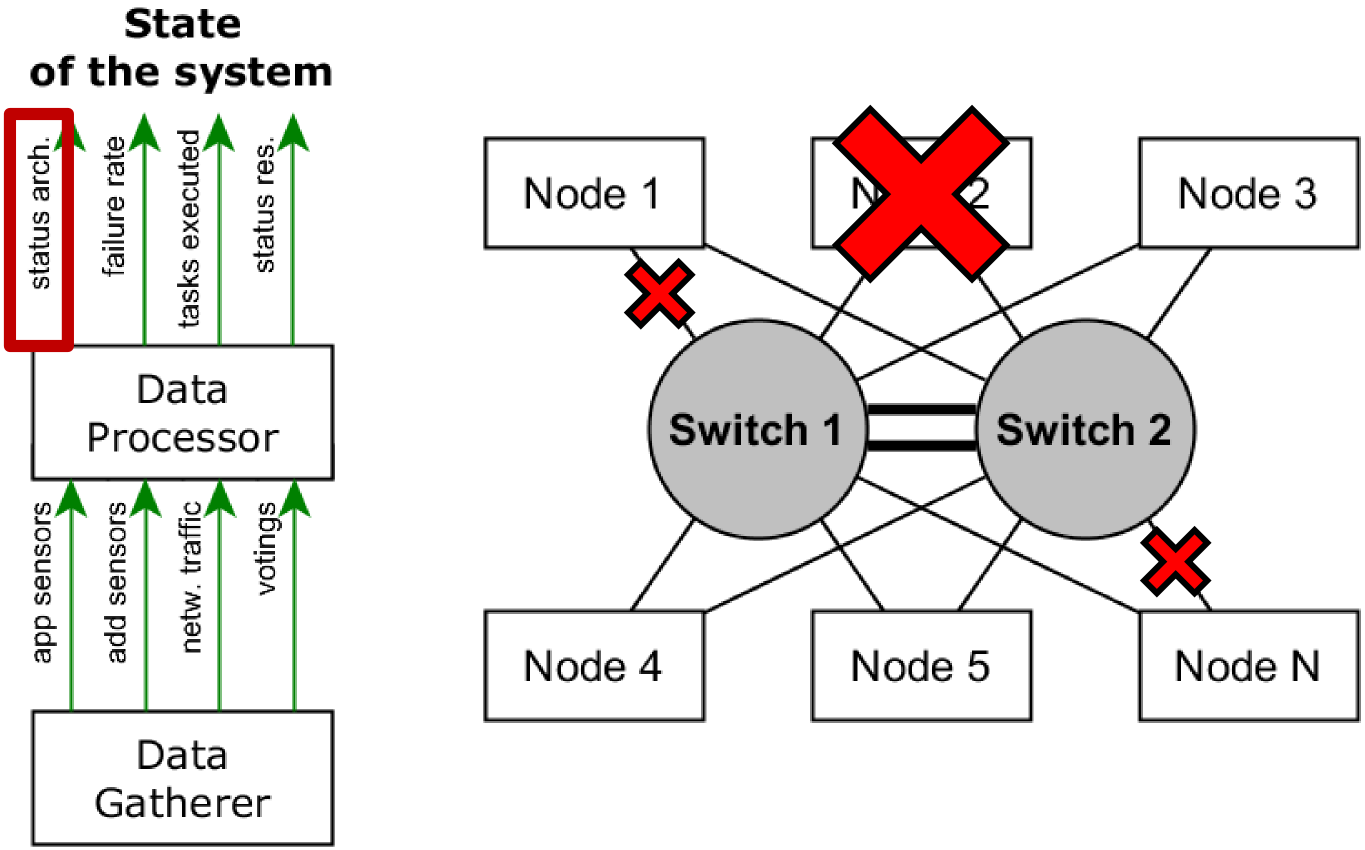




The system architecture

Monitoring Manager

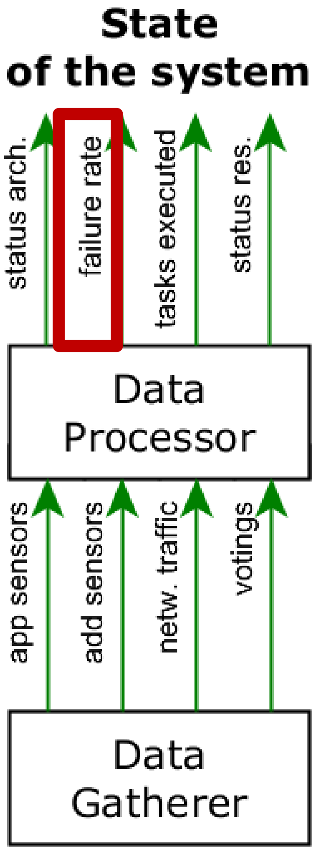
Determine and populate the **state of the system**

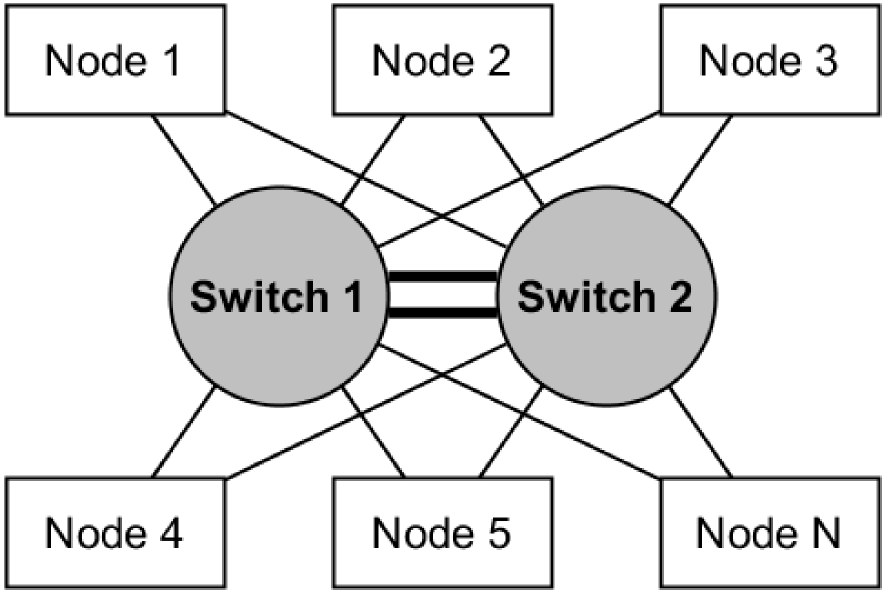


The system architecture

Monitoring Manager

Determine and populate the **state of the system**

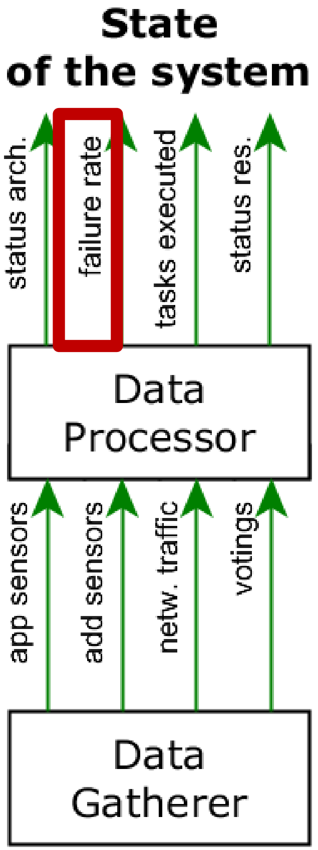


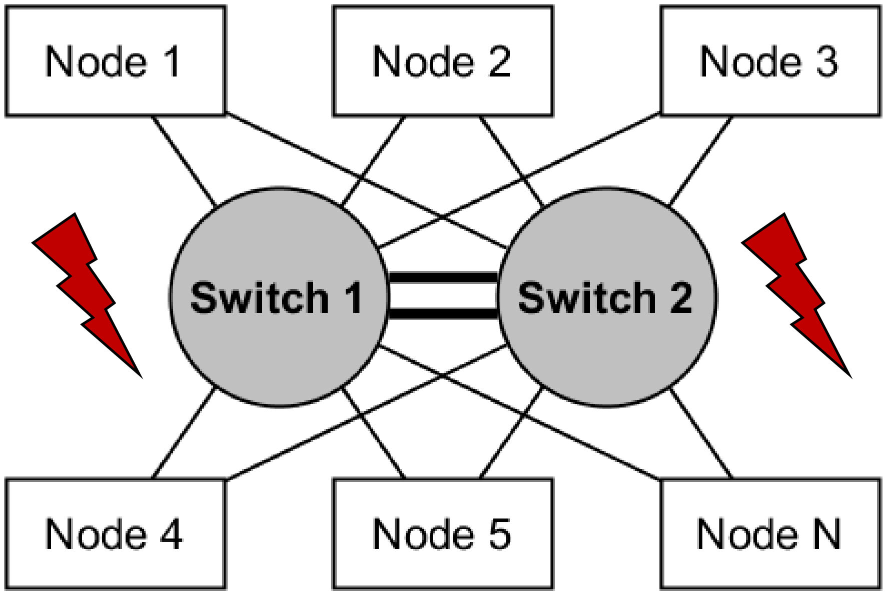


The system architecture

Monitoring Manager

Determine and populate the **state of the system**

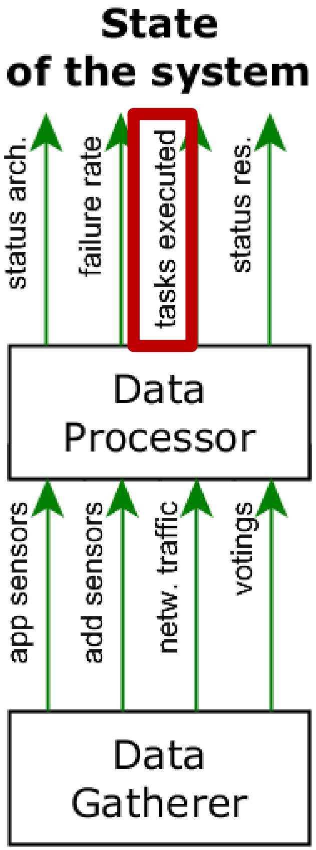


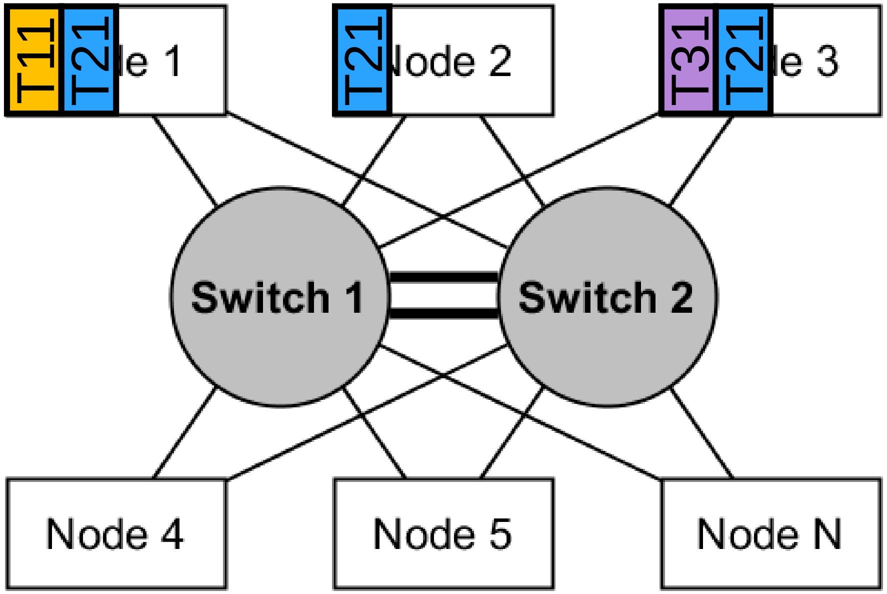


The system architecture

Monitoring Manager

Determine and populate the **state of the system**

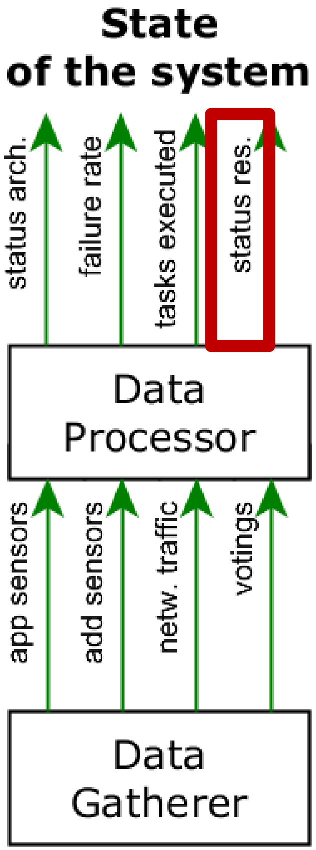


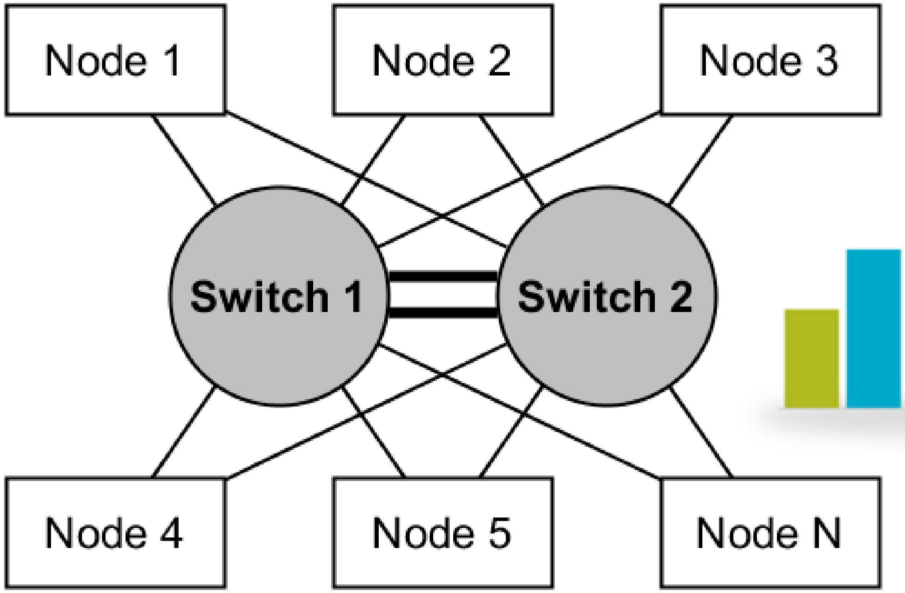


The system architecture

Monitoring Manager

Determine and populate the **state of the system**

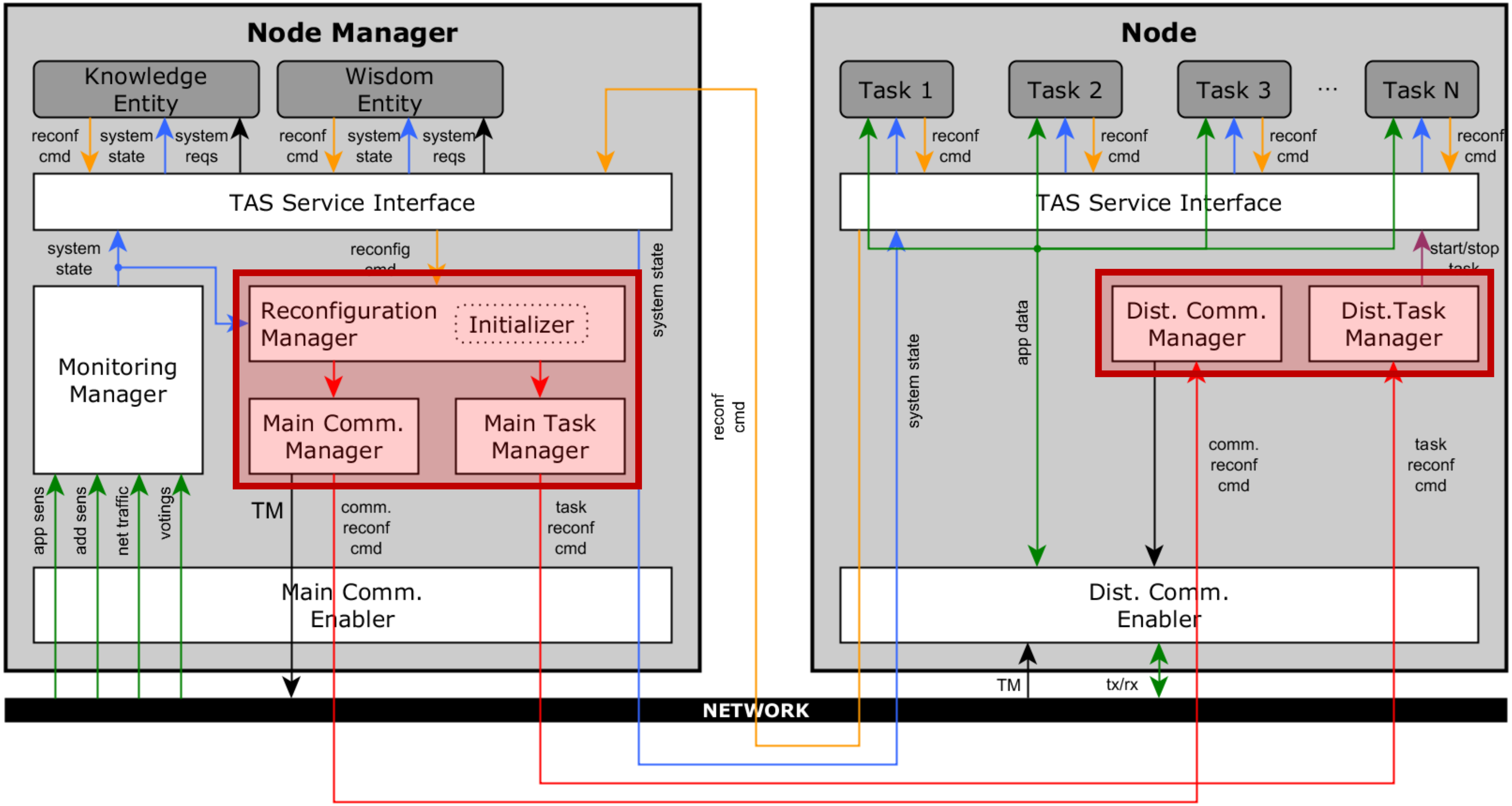




[Node]
CPU
Memory

[Link]
Bandwidth

The system architecture



The system architecture

The reconfiguration process

Possible reconfiguration commands

- start/stop app
- start/stop task
- modify real-time and comm. attributes

1. Decide on the **best configuration**

2. Orchestrate the **reconfiguration process**

The system architecture

The reconfiguration process

1. Decide on the best configuration

- Where to allocate the tasks?
- Consider
  - Resource restrictions
  - Fulfil RT and R(t) reqs
  - Minimize number of changes
- Policies
  - Load balancing (max throughput, min response time and avoid overload of a single component)
  - QoS and QoC
  - Performance of the network
  - Health
  - Energy consumption

The system architecture

The reconfiguration process

1. Decide on the best configuration

- Where to allocate the tasks?
- Consider
  - Resource restrictions
  - Fulfil RT and R(t) reqs
  - **Minimize number of changes → Minimize reconf time**
- Policies
  - Load balancing (max throughput, min response time and avoid overload of a single component)
  - QoS and QoC
  - Performance of the network
  - Health
  - Energy consumption

The system architecture

The reconfiguration process

1. Decide on the best configuration

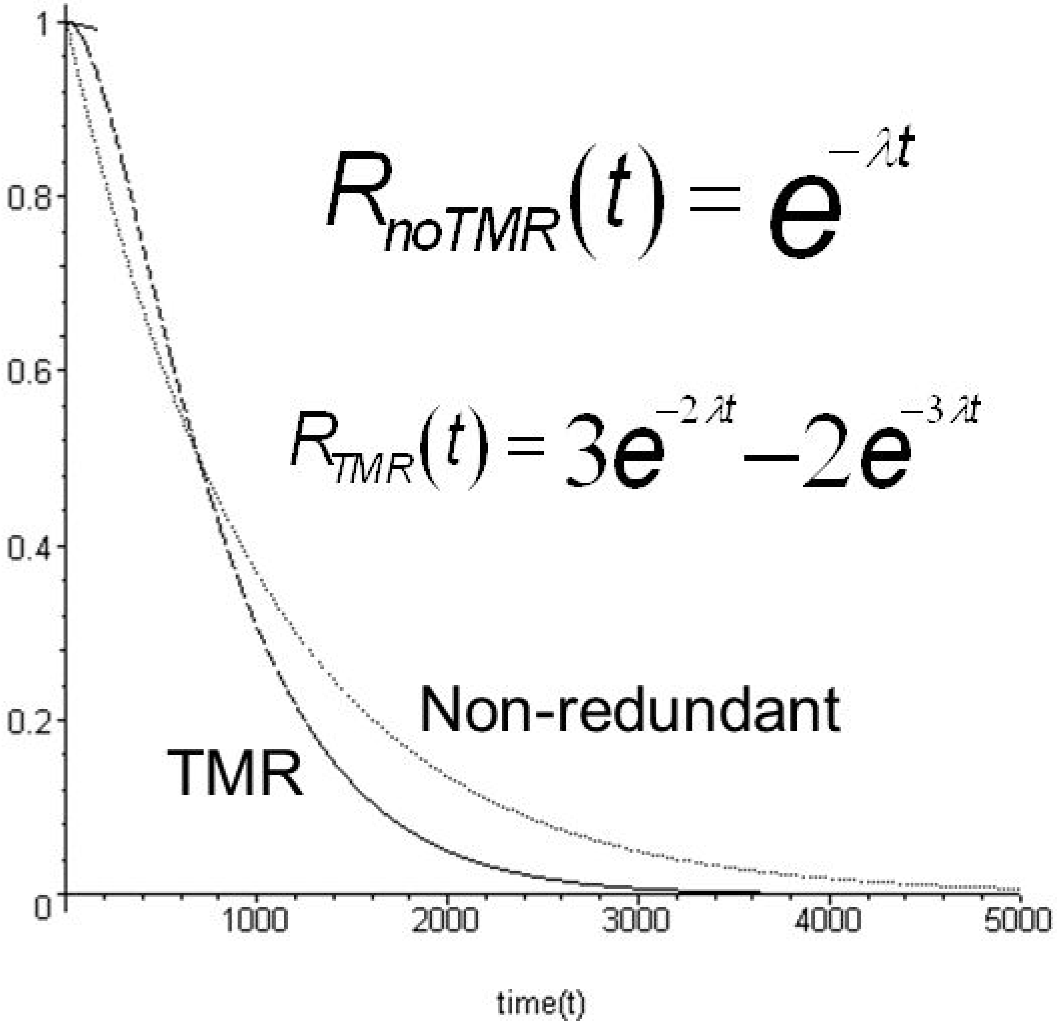
- Where to allocate the tasks?
- Consider
  - Resource restrictions
  - Fulfil RT and R(t) reqs
  - **Minimize number of changes → Minimize reconf time**
- Policies
  - Give resources as fast as possible
  - Try to find the best configuration while the system is running
    - Specific policy (already mentioned)
    - Fault tolerance

The system architecture

The reconfiguration process

1. Decide

- Where to a
- Consider
  - Resou
  - Fulfil
  - **Minim f time**
- Policies
  - Give r
  - Try to s running
    - Sp
    - Fa



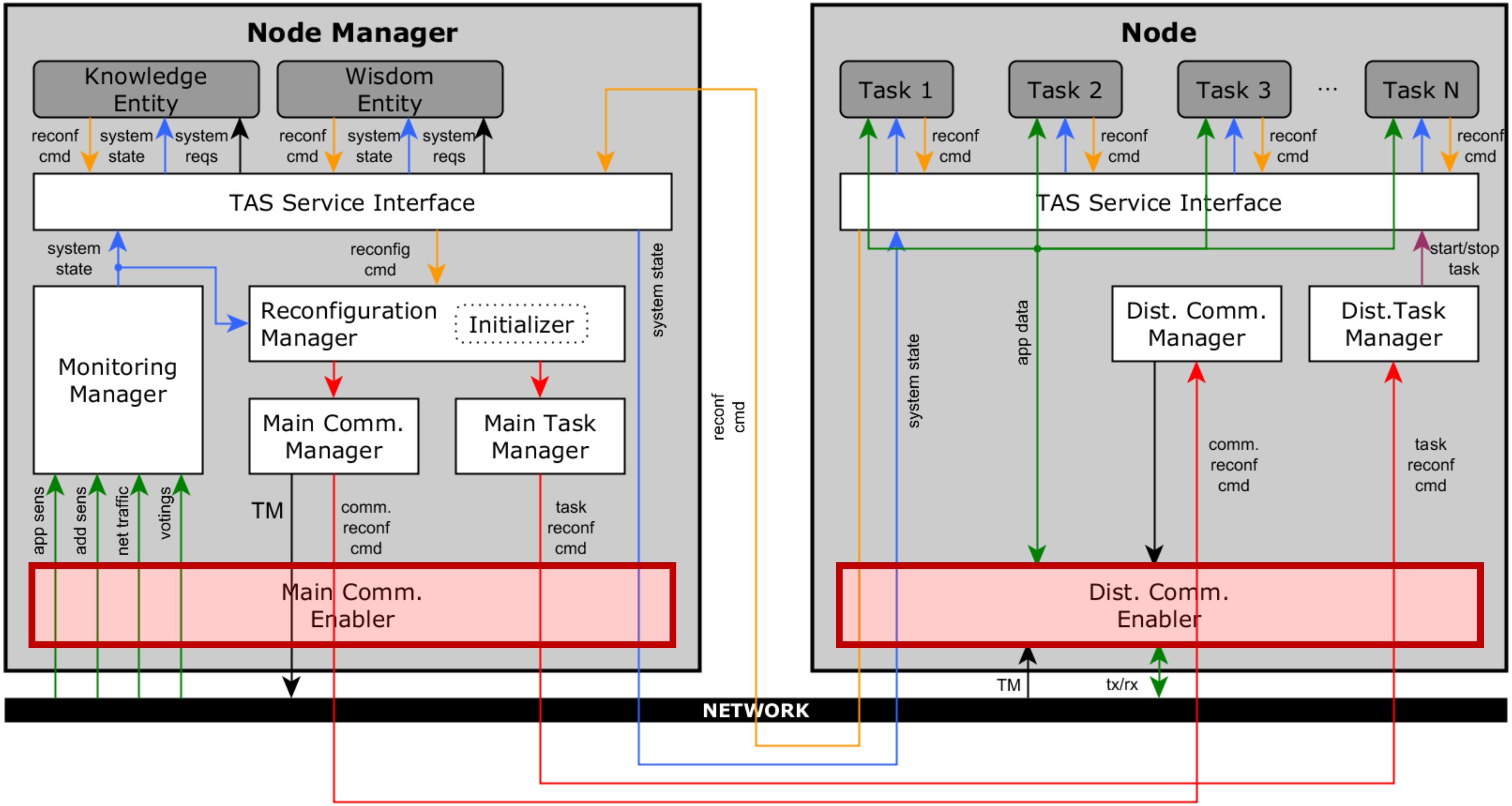
The system architecture

The reconfiguration process

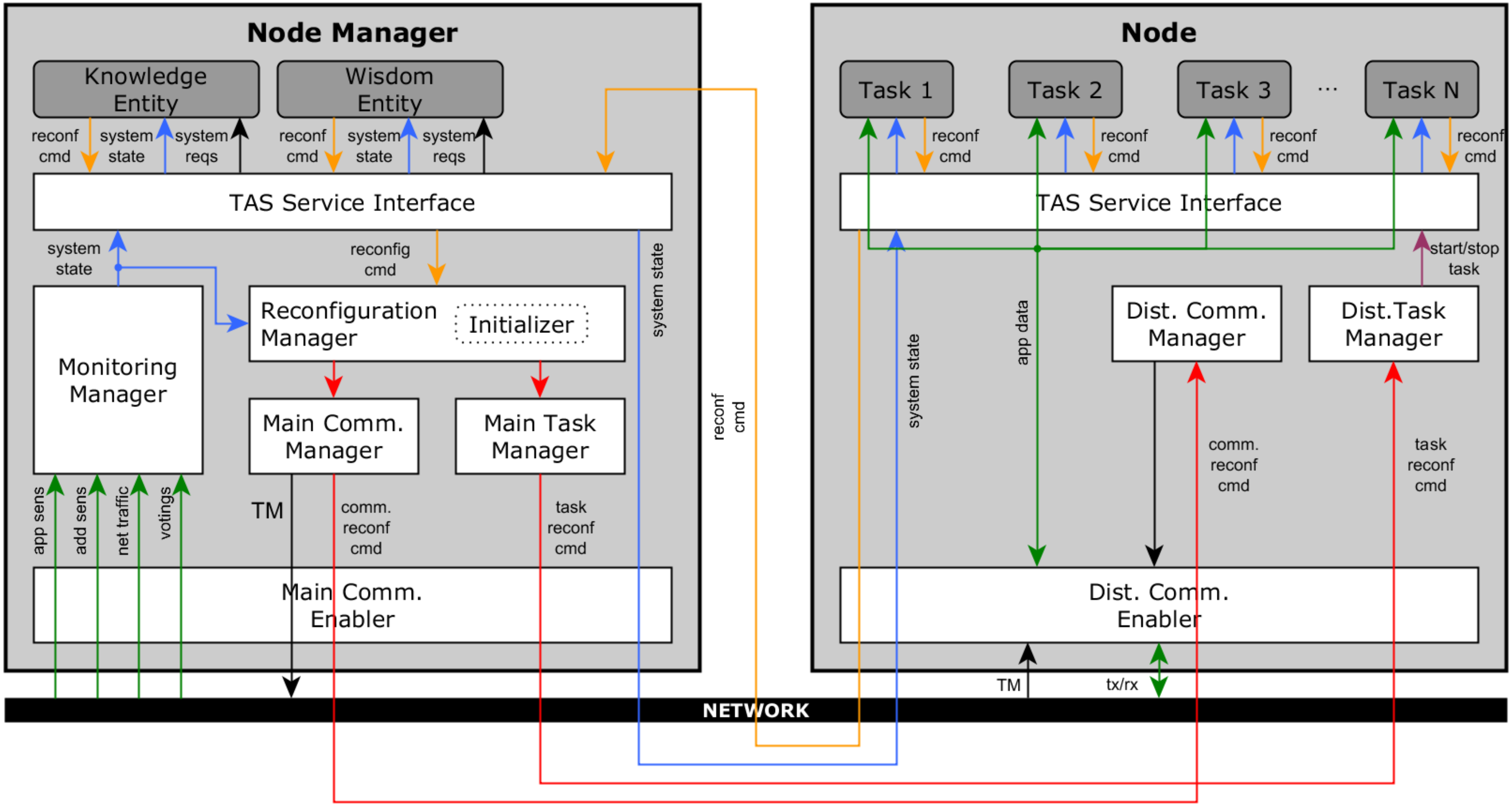
2. **Orchestrate** the **reconfiguration process**

- Generate **ordered set of comm** and **task reconfiguration cmds**
- It can take several ECs

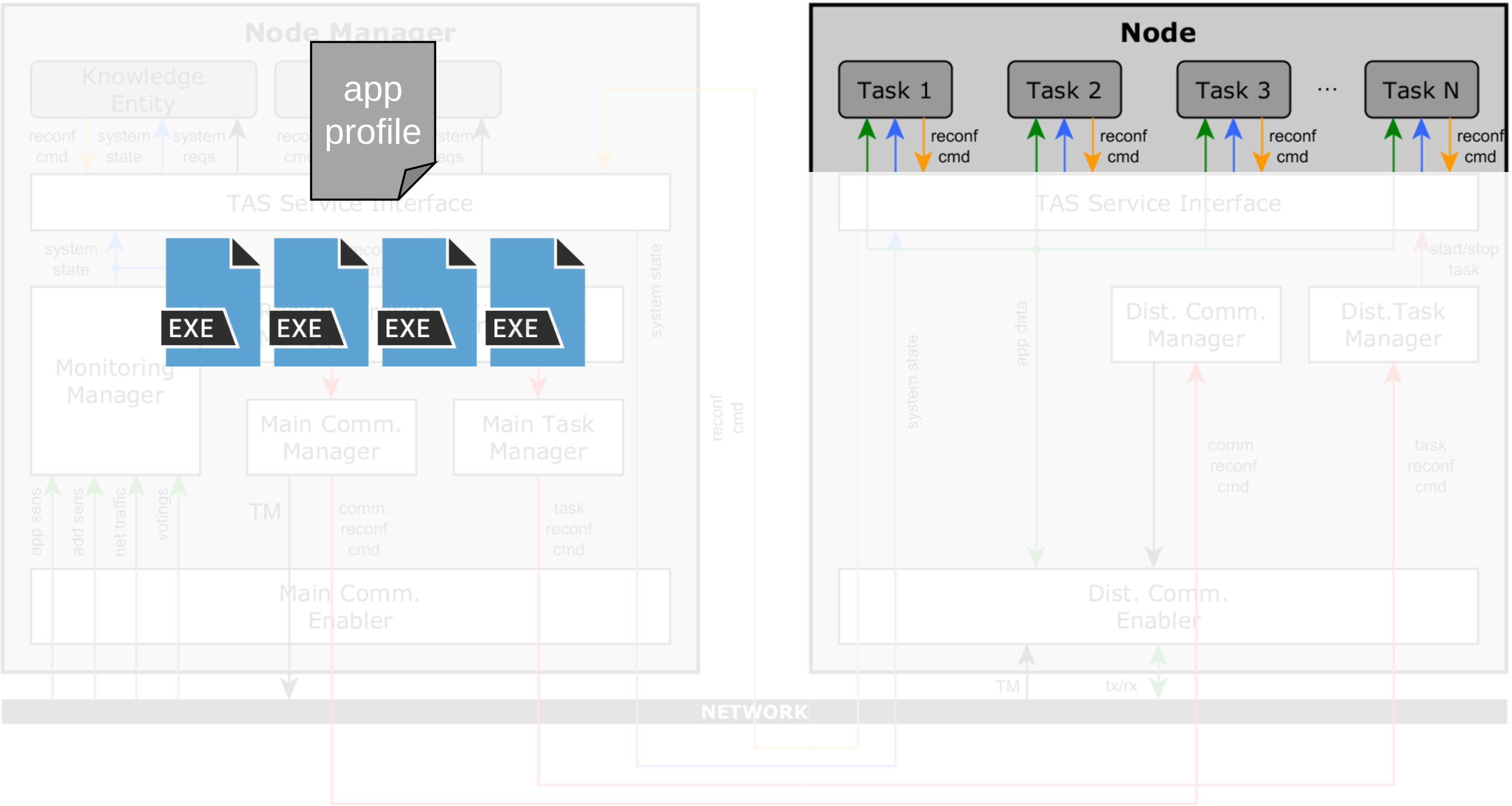
The system architecture



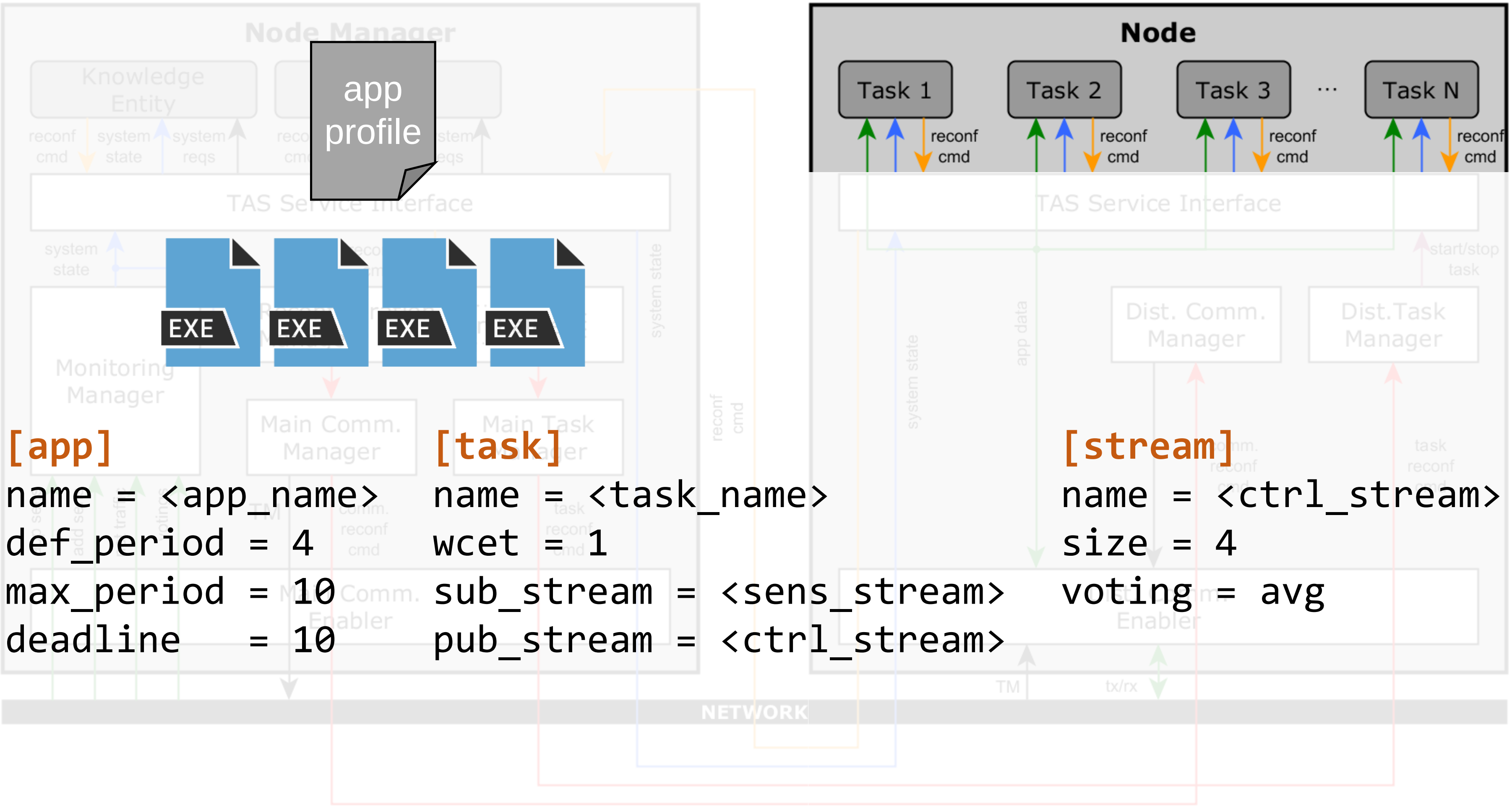
The system architecture



The system architecture



The system architecture

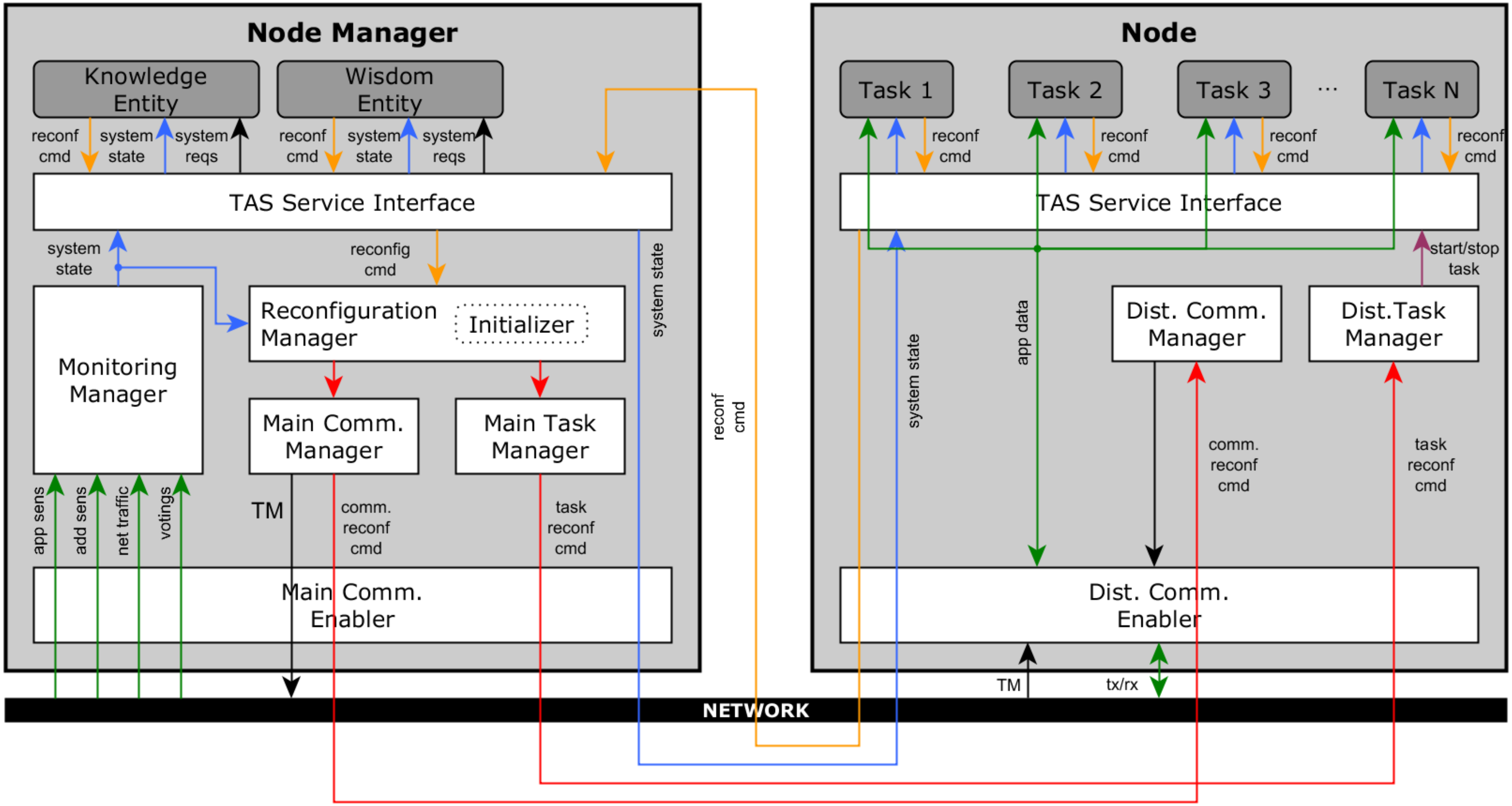


```
[app]
name = <app_name>
def_period = 4
max_period = 10
deadline   = 10
```

```
[task]
name = <task_name>
wcet = 1
sub_stream = <sens_stream>
pub_stream = <ctrl_stream>
```

```
[stream]
name = <ctrl_stream>
size = 4
voting = avg
```

The system architecture



Outline

1. Motivation
2. The problem
3. The task model
4. The system architecture
5. **The Knowledge Entity**
6. The Wisdom Entity

The Knowledge Entity

The KE implements a **rule-based algorithm** that carries out **automatic reconfigurations** to fulfil the **operational requirements** of the system

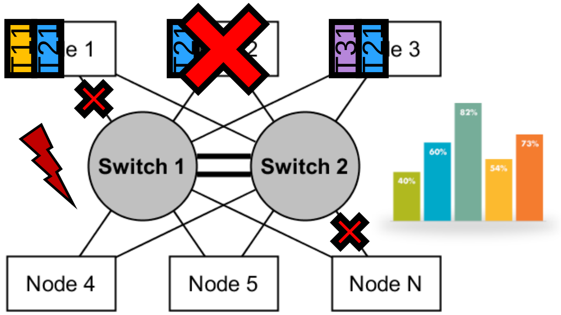
Operational reqs: **List of tasks**, with their **RT and R(t) requirements**, that must be executed

- Indispensable tasks of the phase
- Tasks triggered by human instructions

The Knowledge Entity

State of the system → **Fulfilment?** ← Requirements of the system

Fulfilment? → **Reconfigure**



List of tasks

RT requirements
R(t) requirements

The Knowledge Entity

Discrepancies

List of tasks: change of phase, human cmd or faults

- Stop the tasks that are not needed
- Start the tasks that are needed

RT attributes: change of QoS/QoC requirements

- Reschedule tasks and communications

R(t) attributes: change of R(t) requirements, change of environment (FR/BER) or faults

- Stop the task replicas that are not needed
- Start new task replicas

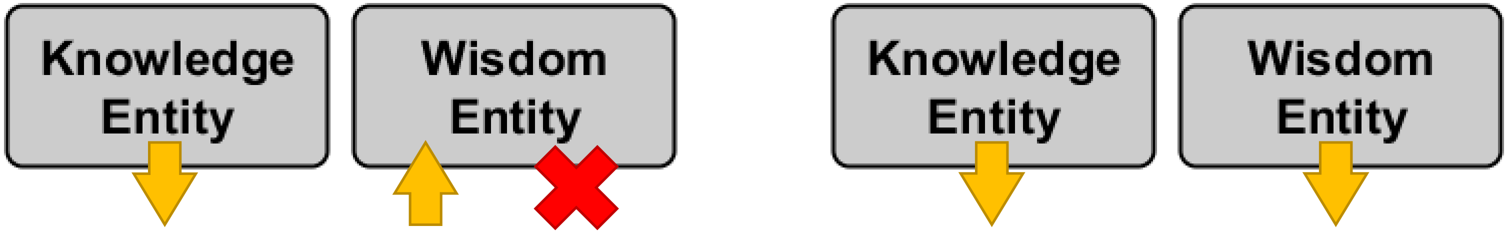
Outline

1. Motivation
2. The problem
3. The task model
4. The system architecture
5. The Knowledge Entity
6. **The Wisdom Entity**

The Wisdom Entity

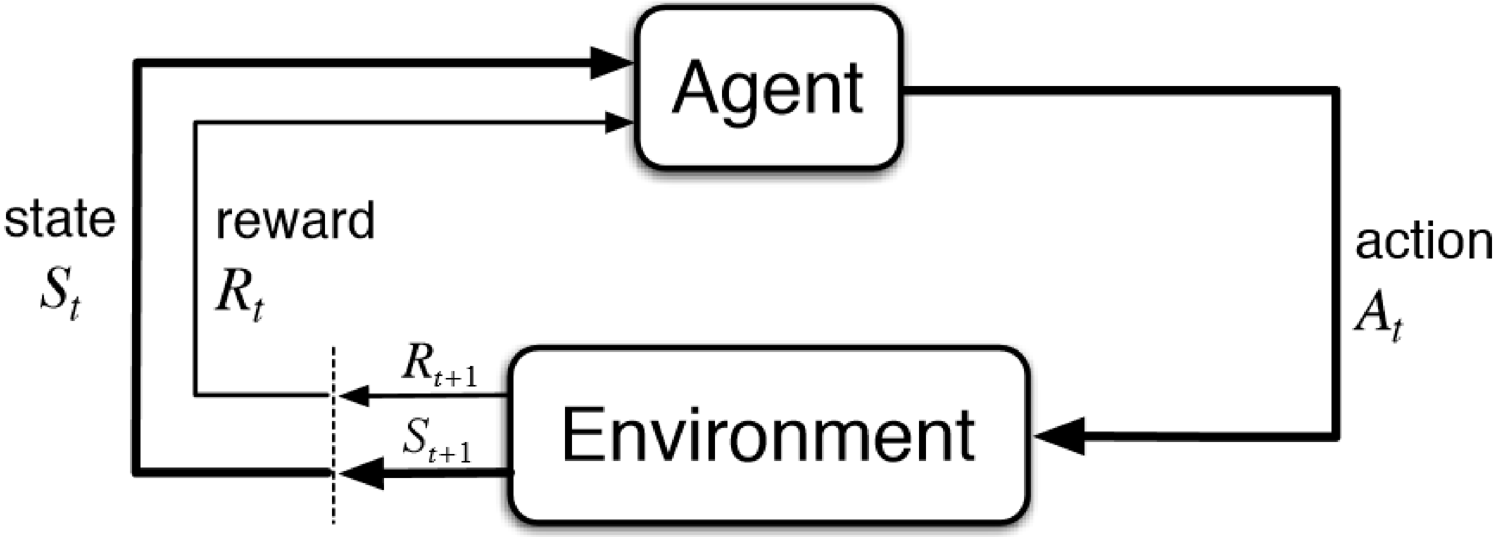
The WE implements a **learning algorithm**, based on **Reinforcement Learning** (RL), that **improves** the **automatic decisions** of the KE

- Medium/large-term decisions
  - Predict future needs to achieve better reactivity
- Infer the best configuration in situations beyond the regular ones



The Wisdom Entity

Reinforcement Learning (RL)



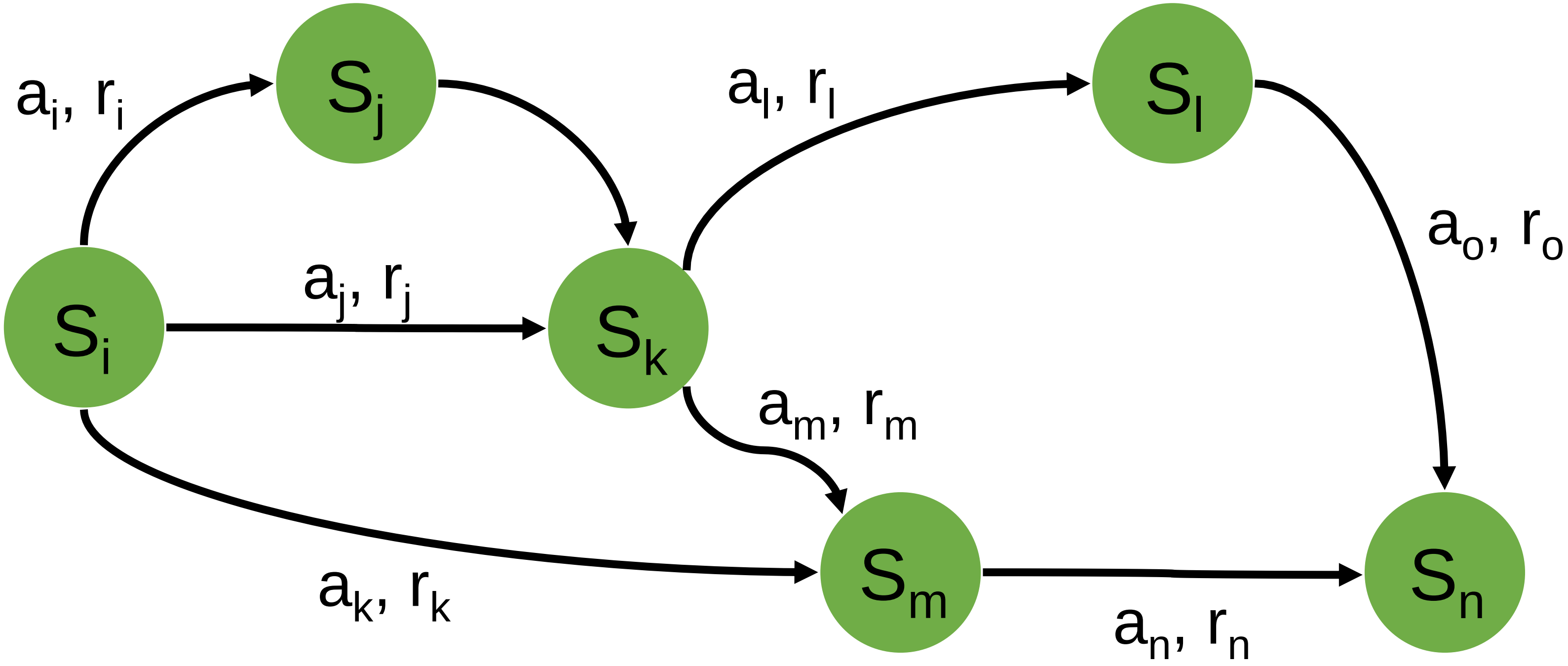
The Wisdom Entity

$S = \{s_1, s_2, \ldots, s_n\}$, set of states

$R(s, a)$, reward

$A = \{a_1, a_2, \ldots, a_m\}$, set of actions

$T(s, a, s') = \Pr(s' | s, a)$, transitions



The Wisdom Entity

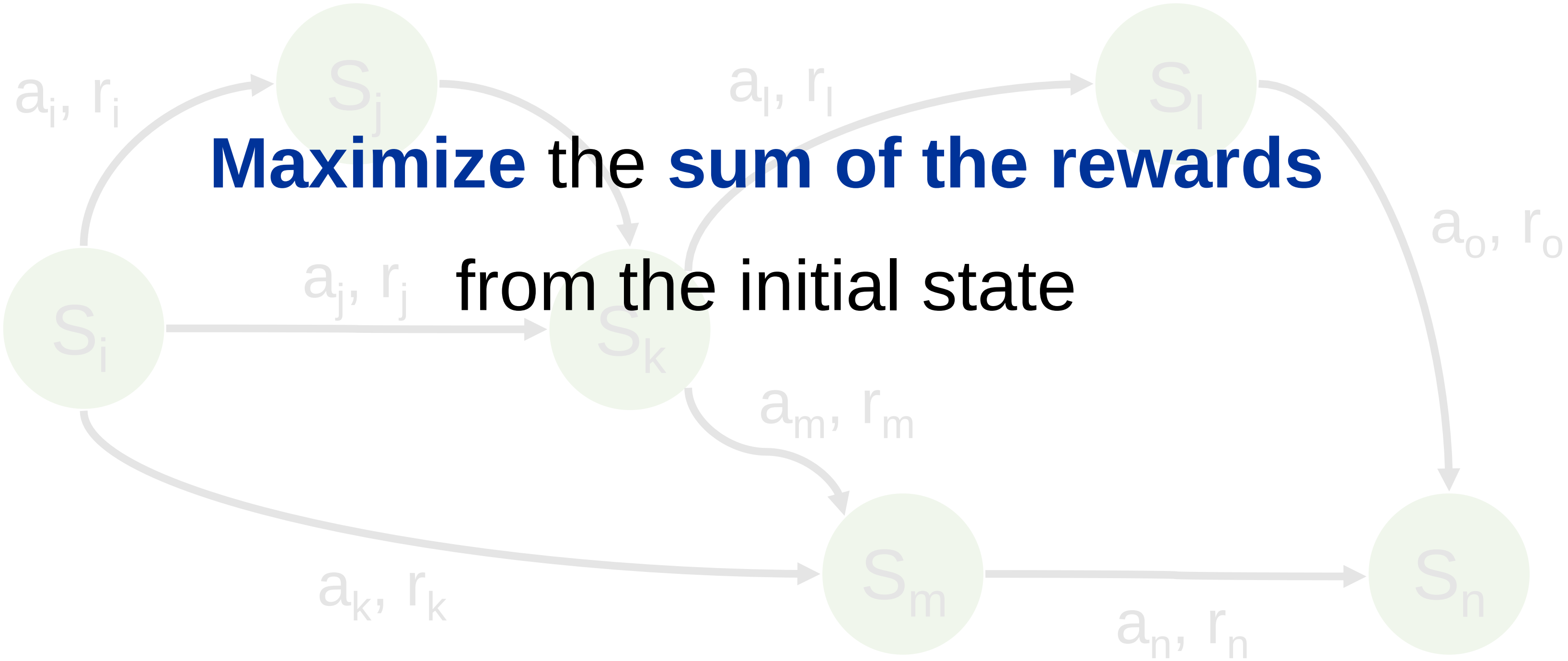
$S = \{s_1, s_2, \dots, s_n\}$, set of states

$R(s, a)$, reward

$A = \{a_1, a_2, \dots, a_m\}$, set of actions

$T(s, a, s') = P(s'|s, a)$, transitions

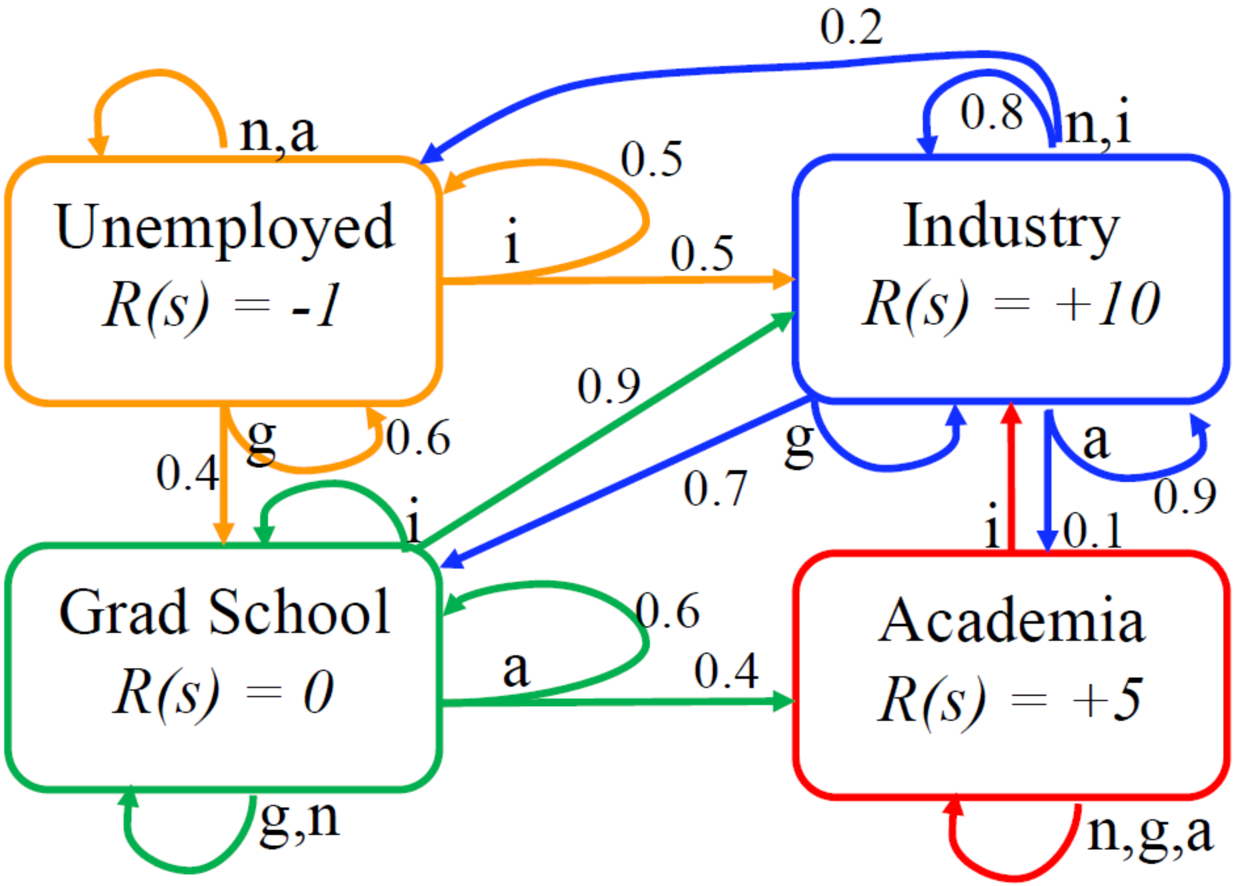
$$argmax_{\pi} E_{\pi}[r_0 + r_1 + \cdots + r_T | s_0]$$



Maximize the **sum of the rewards**

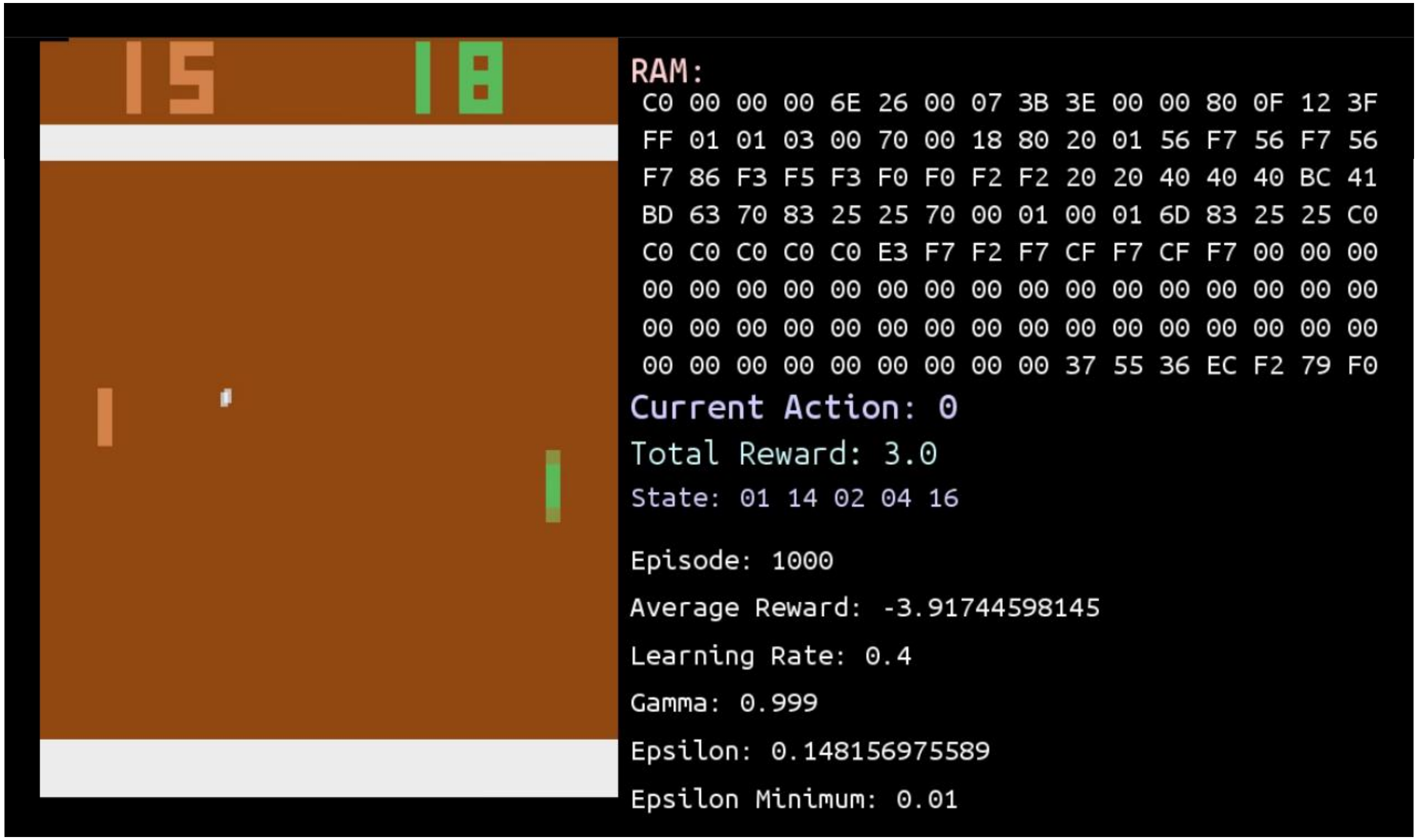
from the initial state

The Wisdom Entity



n = do nothing
i = apply to industry
g = apply to grad school
a = apply to academia

The Wisdom Entity



```
RAM:
 C0 00 00 00 6E 26 00 07 3B 3E 00 00 80 0F 12 3F
 FF 01 01 03 00 70 00 18 80 20 01 56 F7 56 F7 56
 F7 86 F3 F5 F3 F0 F0 F2 F2 20 20 40 40 40 BC 41
 BD 63 70 83 25 25 70 00 01 00 01 6D 83 25 25 C0
 C0 C0 C0 C0 C0 E3 F7 F2 F7 CF F7 CF F7 00 00 00
 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00 00 37 55 36 EC F2 79 F0
Current Action: 0
Total Reward: 3.0
State: 01 14 02 04 16

Episode: 1000
Average Reward: -3.91744598145
Learning Rate: 0.4
Gamma: 0.999
Epsilon: 0.148156975589
Epsilon Minimum: 0.01
```

https://www.youtube.com/watch?v=PSQt5KGv7Vk

The Wisdom Entity

$\boldsymbol{S = \{s_1, s_2, \dots, s_n\}}$, set of states

- State of the system
- Requirements of the system

$\boldsymbol{A = \{a_1, a_2, \dots, a_m\}}$, set of actions

- Nothing
- Reconfiguration

$\boldsymbol{T(s, a, s') = Pr(s'|s, a)}$, transitions

- Changes in the state of the system
  - reconfiguration, faults, external changes
- Changes in the requirements of the system
  - Phase change, on-demand changes

$\boldsymbol{R(s, a)}$, reward

- Learned/Automatic
- Fulfillment of requirements
  - Yes → +reward
  - No → -reward
- QoS and QoC
  - Reward better QoS/QoC
- Other policies…

The Wisdom Entity

It seems **counter-intuitive** to use RL for **deriving rules** in **new situations**…

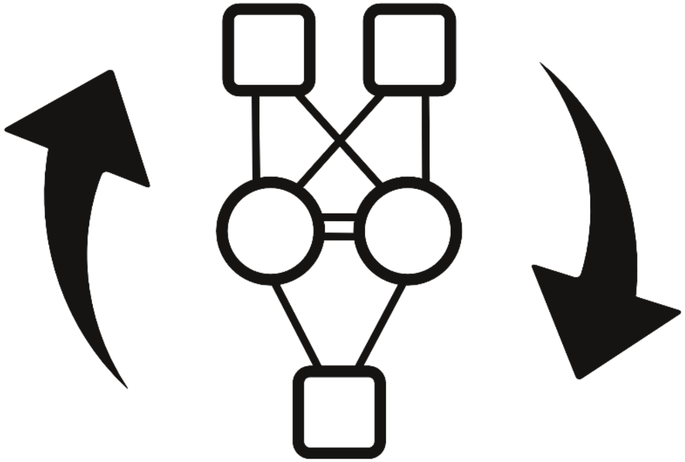
- Take decisions of the non-critical tasks
- Get the rules by off-line simulation

A possible research → compare the performance

- KE (static rules)
- WE (learning alone)
- KE (static rules) + WE (learning from KE)

Metrics: number of changes, QoS, QoC, response time,…

Dynamic Node Replication in the DFT4FTT Architecture



Alberto Ballesteros