Temporal Replication of Messages for Adaptive systems using a Holistic Approach

A. Ballesteros, M. Barranco
S. Arguimbau, M. Costa and J. Proenza
DMI, Universitat de les Illes Balears, Spain
{a.ballesteros, manuel.barranco}@uib.es
{sergi.arguimbau, marc.costa.marquez, julian.proenza}@uib.es

This work was supported by project TEC2015-70313-R (Spanish *Ministerio de Economía y Competividad*) and by FEDER funding.

1. Introduction

Critical **Adaptive Distributed Embedded Systems** (ADES) must meet high **real-time** and **dependability** requirements, while **autonomously rearranging** themselves to operate in **dynamic operational contexts**.


autonomous vehicles

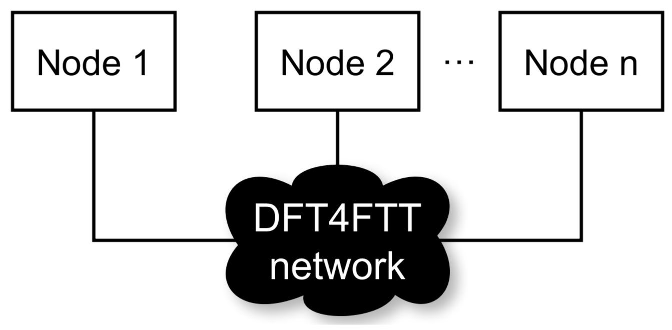

machinery in a smart factory


self-repairing devices

The **DFT4FTT project** proposes a complete **self-reconfigurable infrastructure** for supporting applications with **real-time**, **reliability** and **adaptivity** requirements.

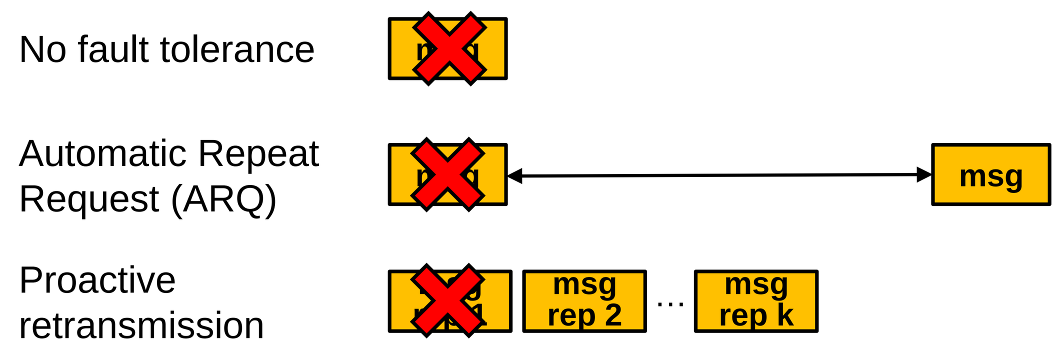
This work addresses the **dynamic temporal replication of messages** used in DFT4FTT to **efficiently tolerate network temporary faults.**

The DFT4FTT Project



- **Network level**: Switched Ethernet impl. of FTT. Real-time, flexibility and dynamic fault tolerance.
- **Node level**: Dynamic allocation of tasks into a set of available computational nodes, while meeting the real-time and reliability requirements.

Tolerating transient faults affecting the network

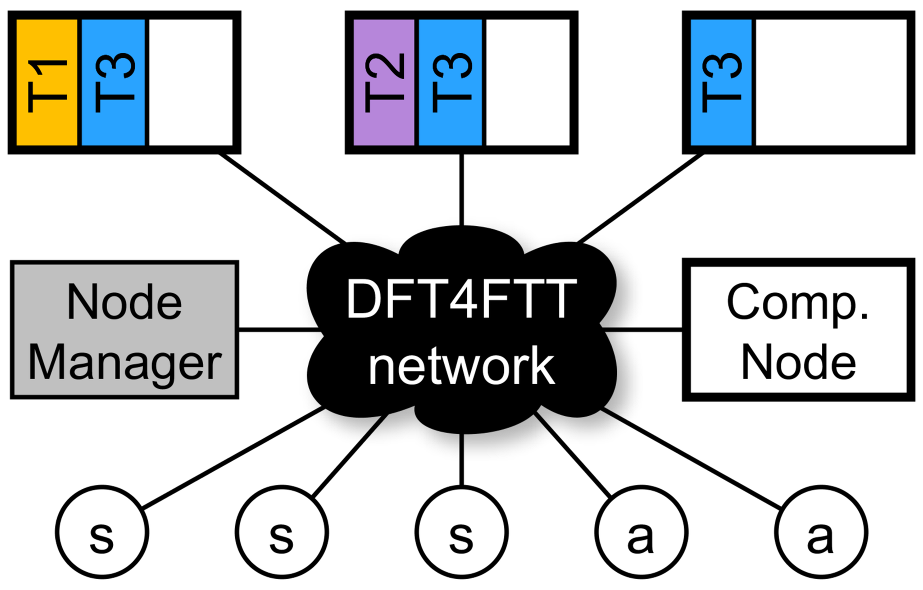


Which value must *k* have?

- **Static** message replication can be **inefficient** or, even, **ineffective**
- **Dynamic** message replication: Change at runtime the number of message replicas (*k*) depending on the current operational context

2. System Architecture

The DFT4FTT architecture is composed of: a **network**, several **sensors and actuators**, several **computational nodes** (CNs) and a **Node Manager** (NM)



The **self-reconf. process** is carried out in **three phases**

Monitoring
Obtain the system state

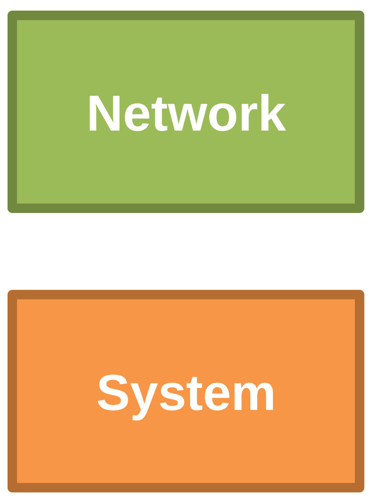
Decision
Determine when and how to switch to a new configuration

Configuration change
Carry out the system modifications

3. Dynamic Message Replication Rationale

Network-level policy: Rely on the network to manage the value of *k*.
- Criticality of the communication
- Available spatial redundancy
- Probability of message loss

System-level policy: The NM manages the value of *k* considering the available FT mechanisms.
- Replication of tasks
- Reintegration of tasks
- Available spatial redundancy
- ….



The **system-level** policy provides the **required level of reliability** in a **cost-effective manner** by **exploiting** all the **fault tolerance mechanisms**

4. Dynamic Message Replication

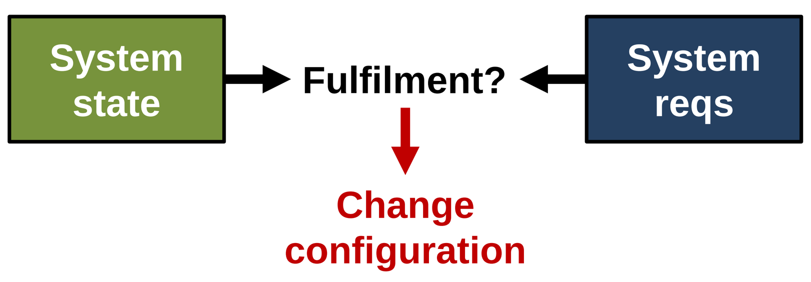
Detection of the need for changes

Identify changes in the **operational context** that require a **change** in the **number of message replicas**

Environment: Temporary faults affecting the network.
- Radiation sensors and failure rate model
- Ethernet card error statistics
- I Am Alive message
- CVEP which involves using ACKs

System itself: Permanent faults affecting the network.
- Above mention (except the first one)

Operational reqs: Changes in the comm. requirements.
- The NM maintains the list of operational requirements



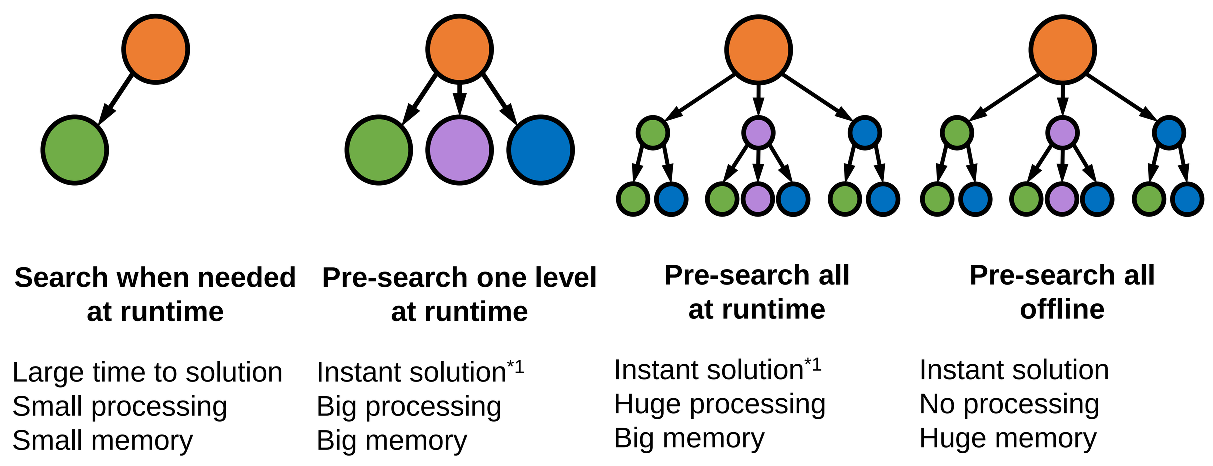
Determination of the new configuration

Search, among all the possible configurations, one that **fulfills** all the **operational requirements** (including *k*)

It must include a **holistic scheduler** and a **reliability analyzer**
- **Heuristic-based techniques**: Branch and bound with a greedy algorithm.
- **Metaheuristic-based techniques**: Tabu search.
- **Solvers**: SMT solvers.

Still, an **on-line search** can require a **huge amount of time**

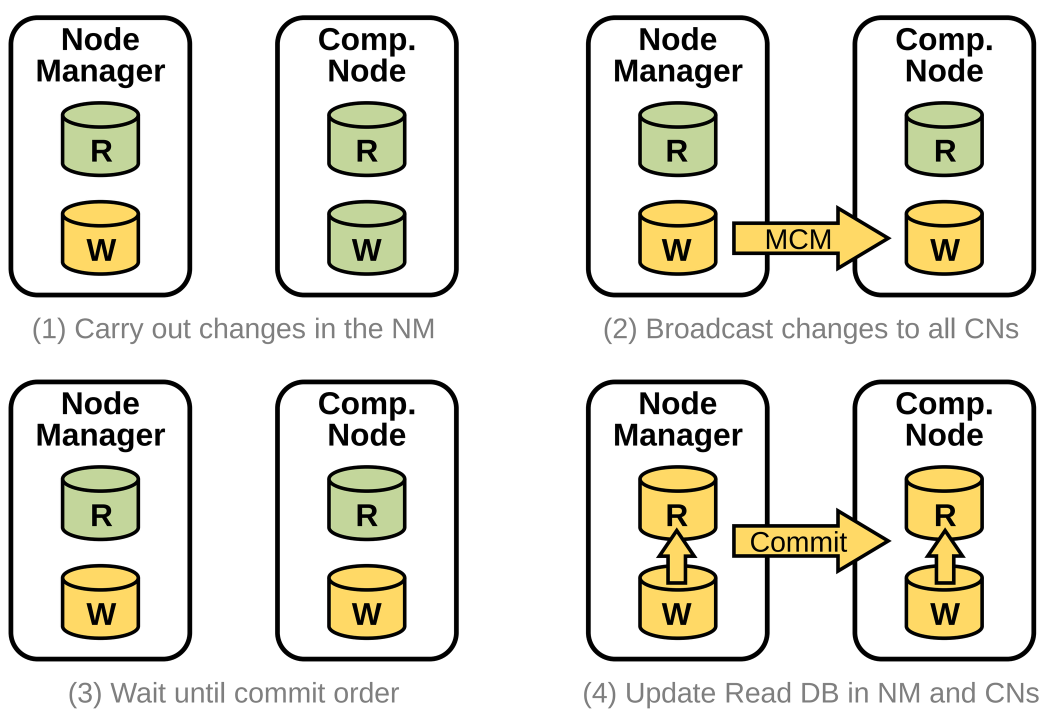
Execute at runtime or completely/partially pre-calculated offline



Propagation of the new configuration

Propagate the value of ***k*** for each comm. **to the CNs**

Design a **mechanism** to **keep comm. DBs consistent**
- **Read DB** and **Write DB** in each network component.
- Two phases: broadcast changes and commit changes.

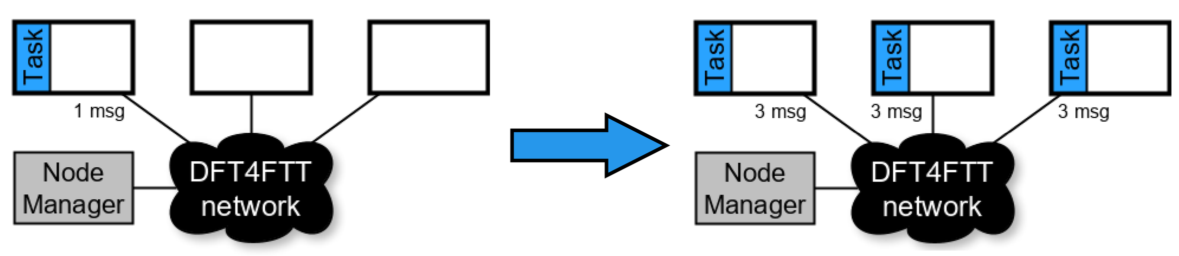


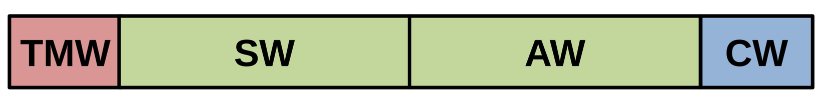
5. Experimentation

We have constructed a **prototype** implementing all the **work presented**, **except** for the **searching mechanism**
- Generate k message replicas
- Modify k at runtime
- Propagate the new k to the CNs

Use this prototype to **test** the **correct operation** of the just-mentioned **mechanisms in conjunction**

Switch between configurations where the value of *k* varies





Elementary Cycle: 20 ms

The test demonstrated that **the mechanisms worked as intended**, thereby **demonstrating** their **feasibility**

i: We instruct the NM to reconfigure the system.

i+1 (AW): The NM broadcasts the changes.

i+1 (CW): The NM and the CNs consolidate their DBs.

i+2: The NM triggers the execution of all the task replicas and the transmission of their messages.

6. Conclusions and on-going Work

Design and **partial implementation** of the DFT4FTT mechanism for the **dynamic replication of messages**.

This **increases** the **tolerance** to **temporary faults** affecting **network** in a **cost-effective** manner by **taking advantage** of the available **fault tolerance mechanisms** at **different levels of the architecture**.

In the short term we will **evaluate** and **select** a **search techniques** for finding **valid system configurations** and preferably in a **bounded time**.

Moreover, we will also **study** how to **include** a **holistic scheduler** and a **reliability analysis**.

DFT4FTT project

24TH INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION
ETFA 2019