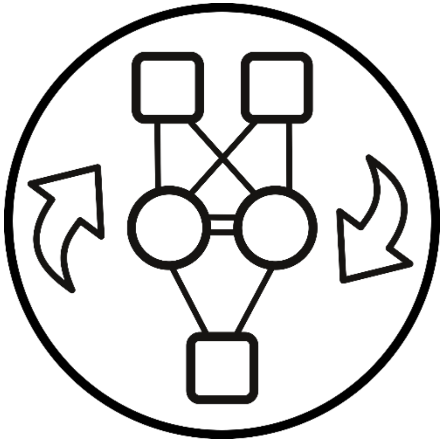
Reconfiguration Strategies for Critical Adaptive Distributed Embedded Systems



Alberto Ballesteros
Julián Proenza
Manuel Barranco
Luís Almeida

UIB
Universitat de les Illes Balears

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD

Fondo Europeo de Desarrollo Regional

Introduction

Adaptive Distributed Embedded Systems (ADES) can **change autonomously** and **dynamically** in response to **unexpected operational requirements** or **conditions**

Introduction

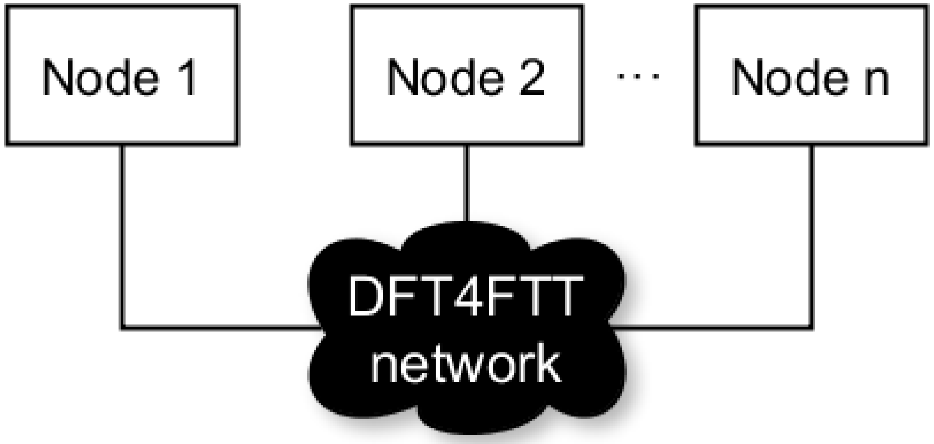
Adaptive Distributed Embedded Systems (ADES) can **change autonomously** and **dynamically** in response to **unexpected operational requirements** or **conditions**

Adaptivity is and interesting feature in terms of:

- **Functionality** → Change the behaviour
- **Efficiency** → Load the necessary functionalities
- **Dependability** → Adaptive fault tolerance

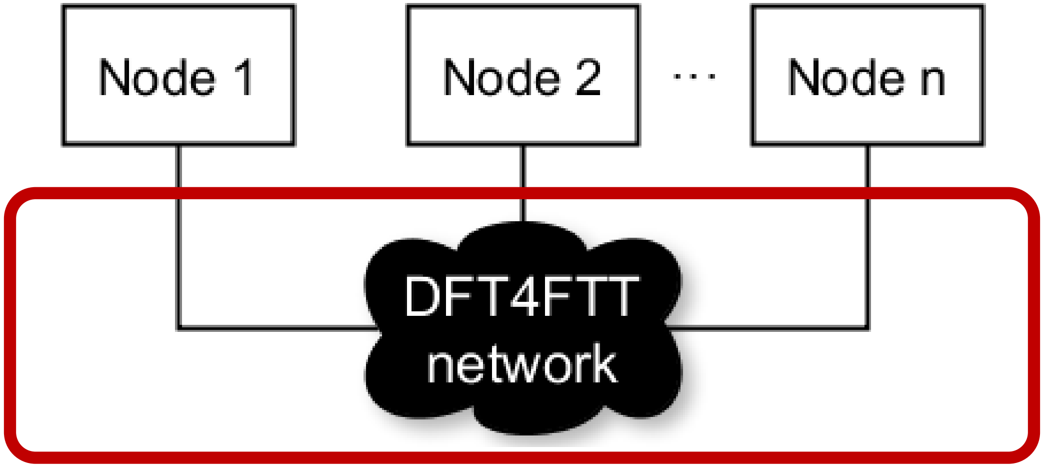
The DFT4FTT project

To **properly implement an ADES** it must be provided with the appropriate **architecture** and **mechanisms**, that make it possible to fulfil its **real-time**, **dependability** and **adaptivity** requirements



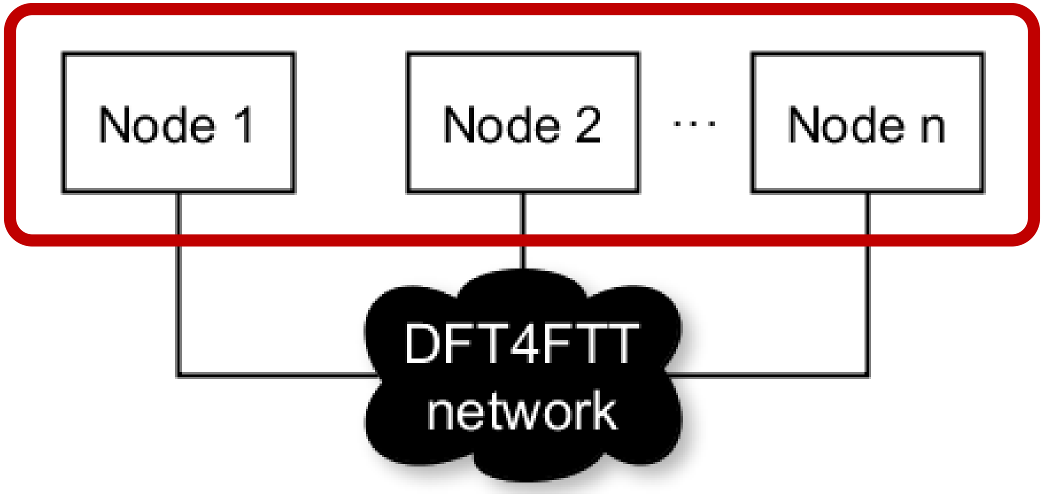
The DFT4FTT project

To **properly implement an ADES** it must be provided with the appropriate **architecture** and **mechanisms**, that make it possible to fulfil its **real-time**, **dependability** and **adaptivity** requirements



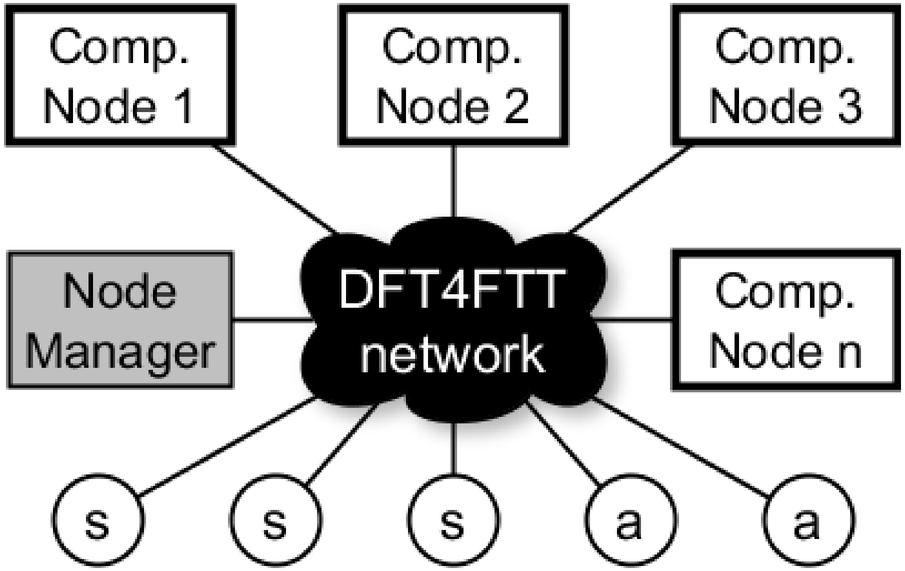
The DFT4FTT project

To **properly implement an ADES** it must be provided with the appropriate **architecture** and **mechanisms**, that make it possible to fulfil its **real-time**, **dependability** and **adaptivity** requirements



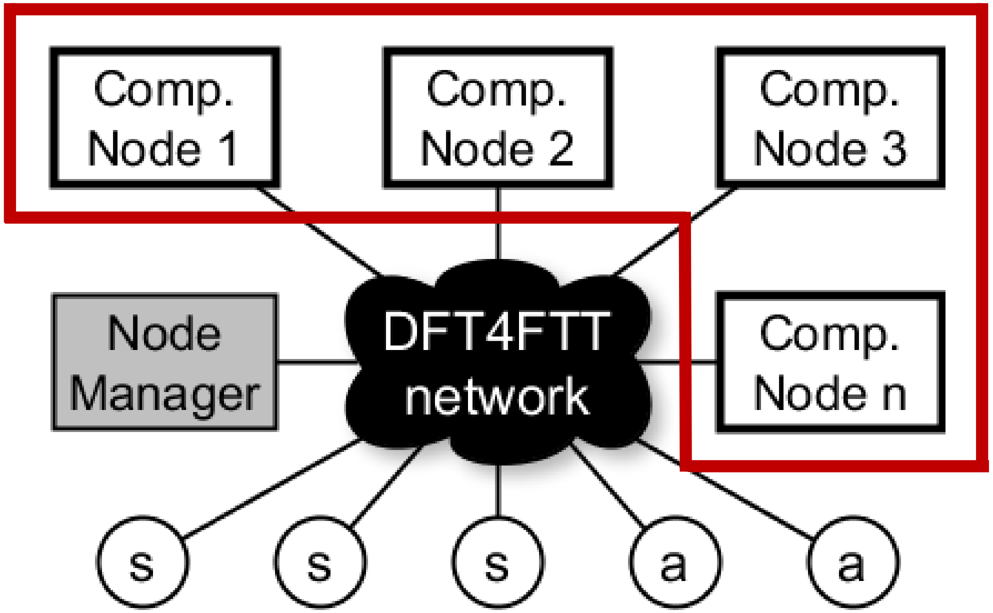
The System Architecture

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



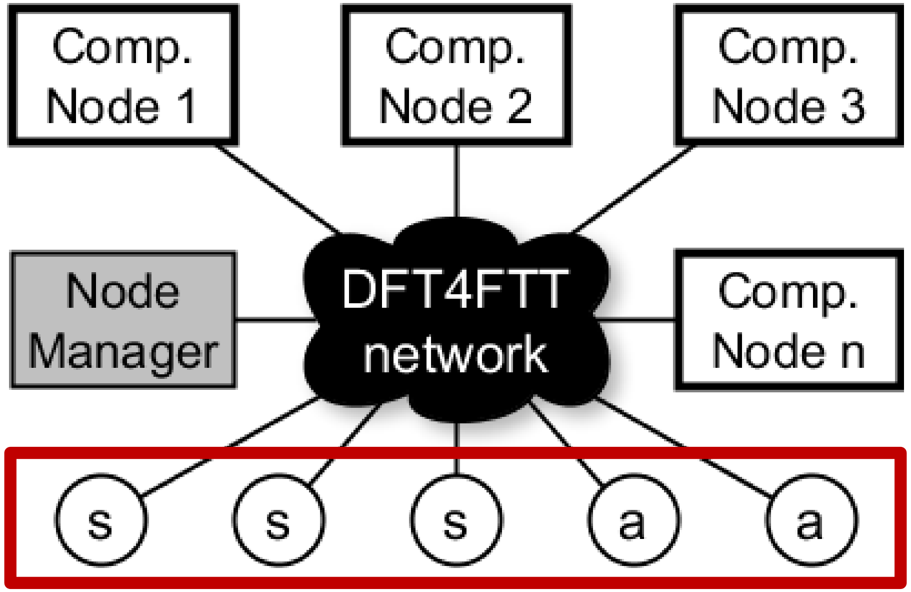
The System Architecture

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



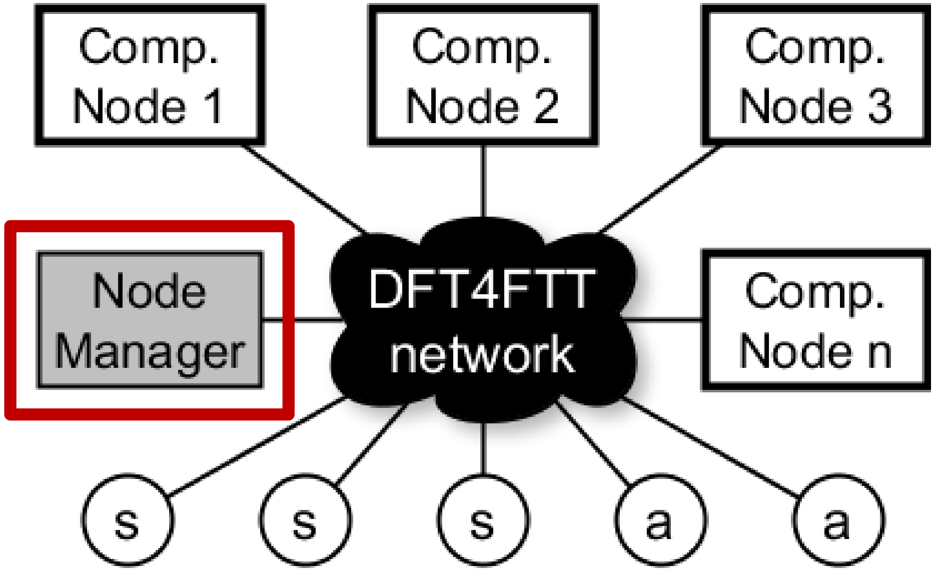
The System Architecture

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



The System Architecture

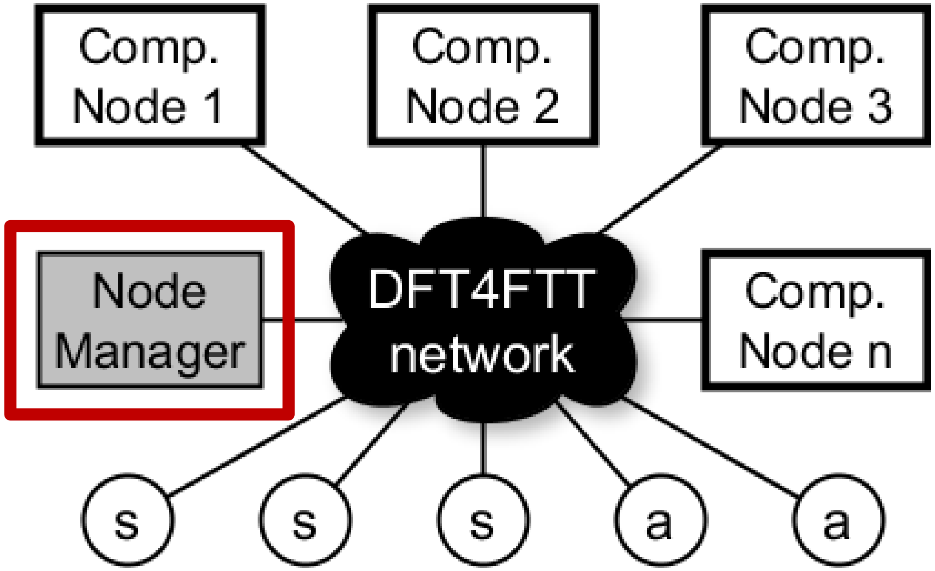
At the **node level**, the DFT4FTT architecture is composed of **various components**



The System Architecture

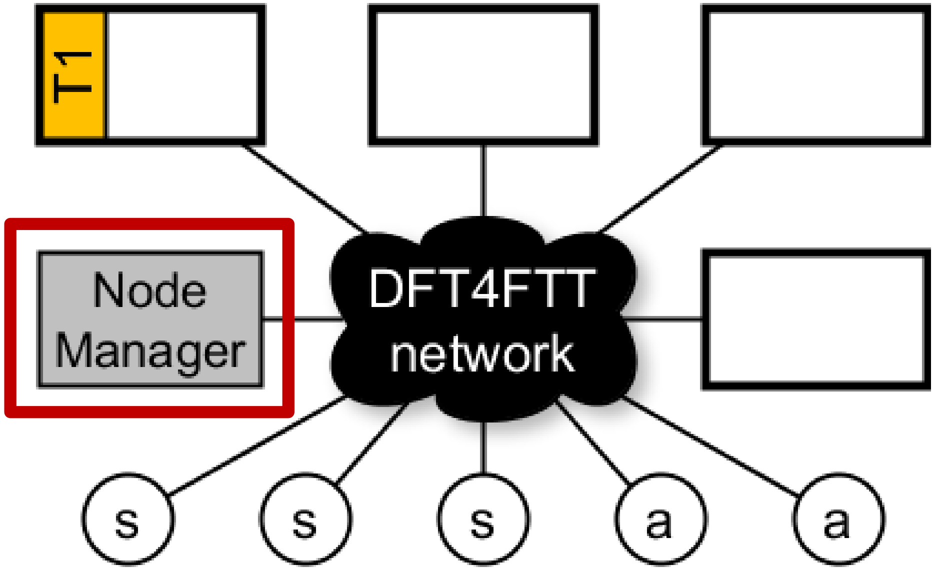
At the **node level**, the DFT4FTT architecture is composed of **various components**

- **Monitor**
- **Detect**
- **Configuration change**



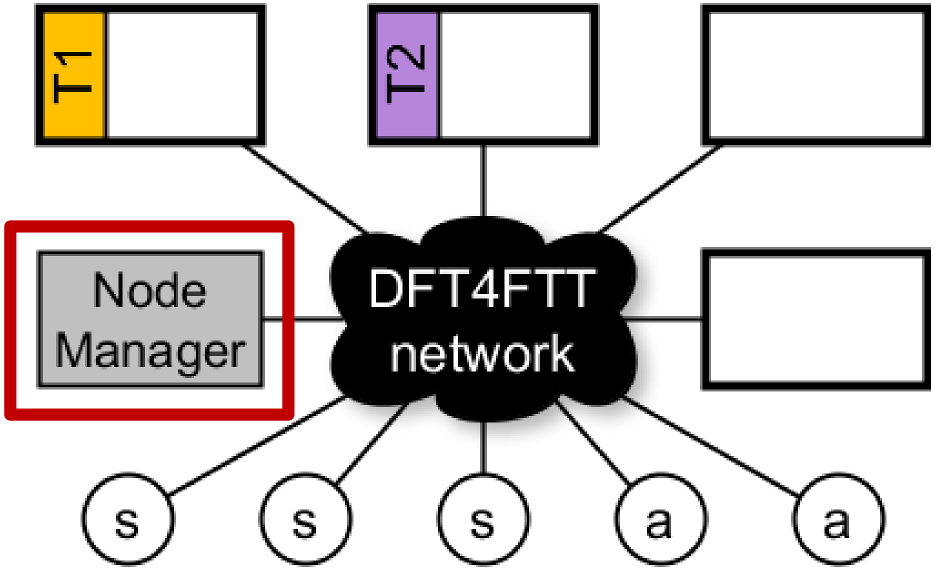
The task allocation scheme

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



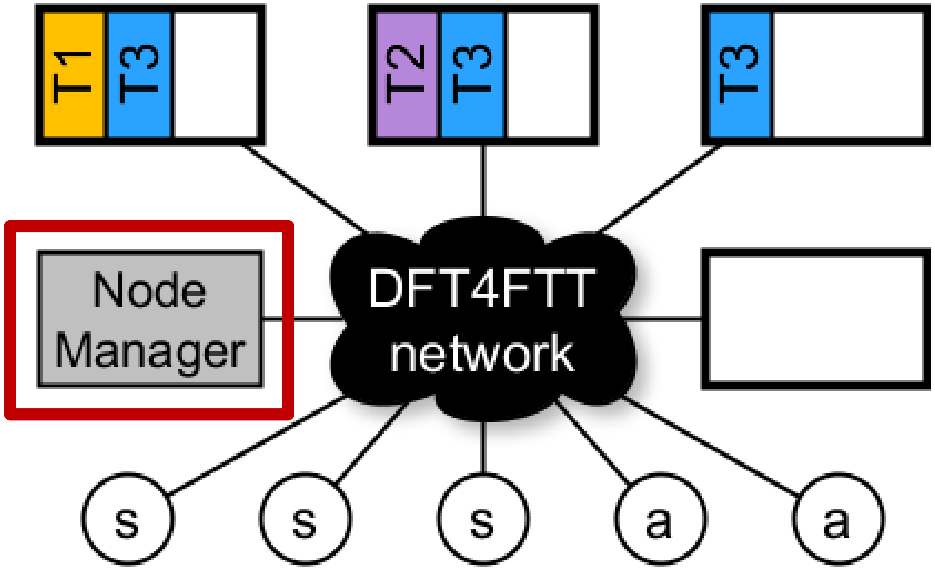
The task allocation scheme

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



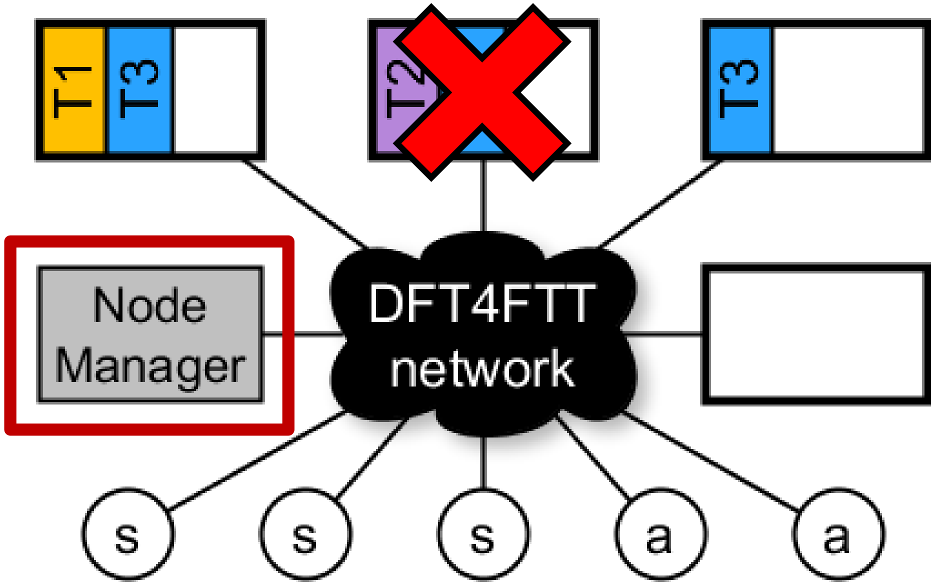
The task allocation scheme

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



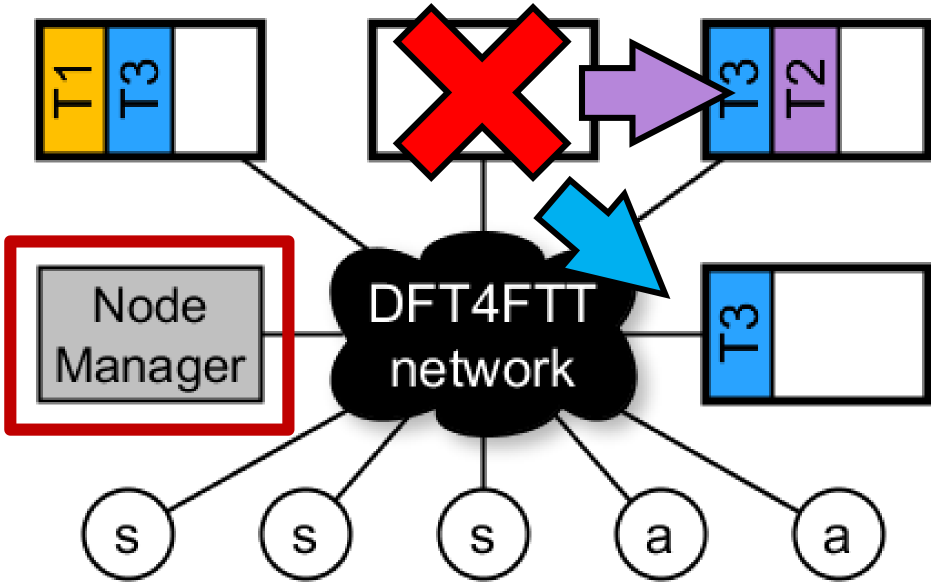
The task allocation scheme

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



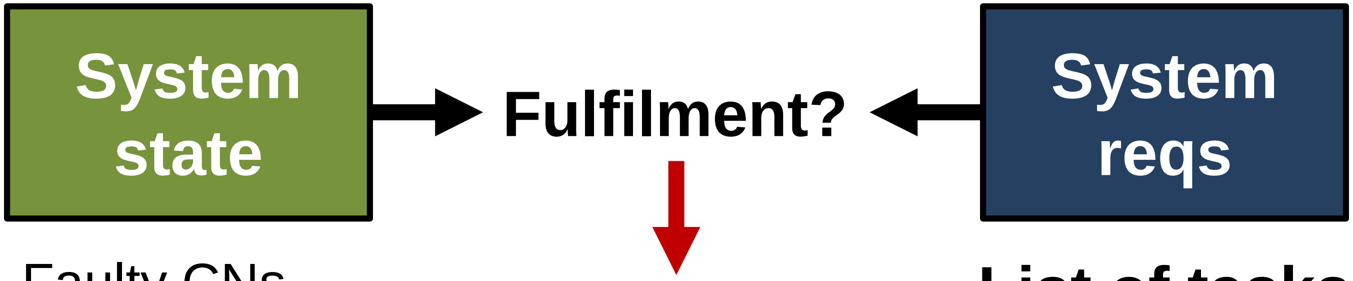
The task allocation scheme

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



Reconfiguration Strategies

Constantly verify that the **system reqs** are **fulfilled**



- Faulty CNs
- Tasks executed
- Falure rates
- …

Change configuration

List of tasks

RT requirements
R(t) requirements

Reconfiguration Strategies

Reliability perspective

The **reconfiguration capabilities** of the NM allows us to **reallocate** the **tasks** being executed in one CN to another, when the first one suffers a **permanent failure**.

Non-critical tasks

- The service is restores after some downtime.

Critical (replicated) tasks

- We have redundancy preservation.
- Equivalent to N-Modular Redund. scheme with spares.

Reconfiguration Strategies

Thank you for your attention

See you at the poster session!

Reliability p

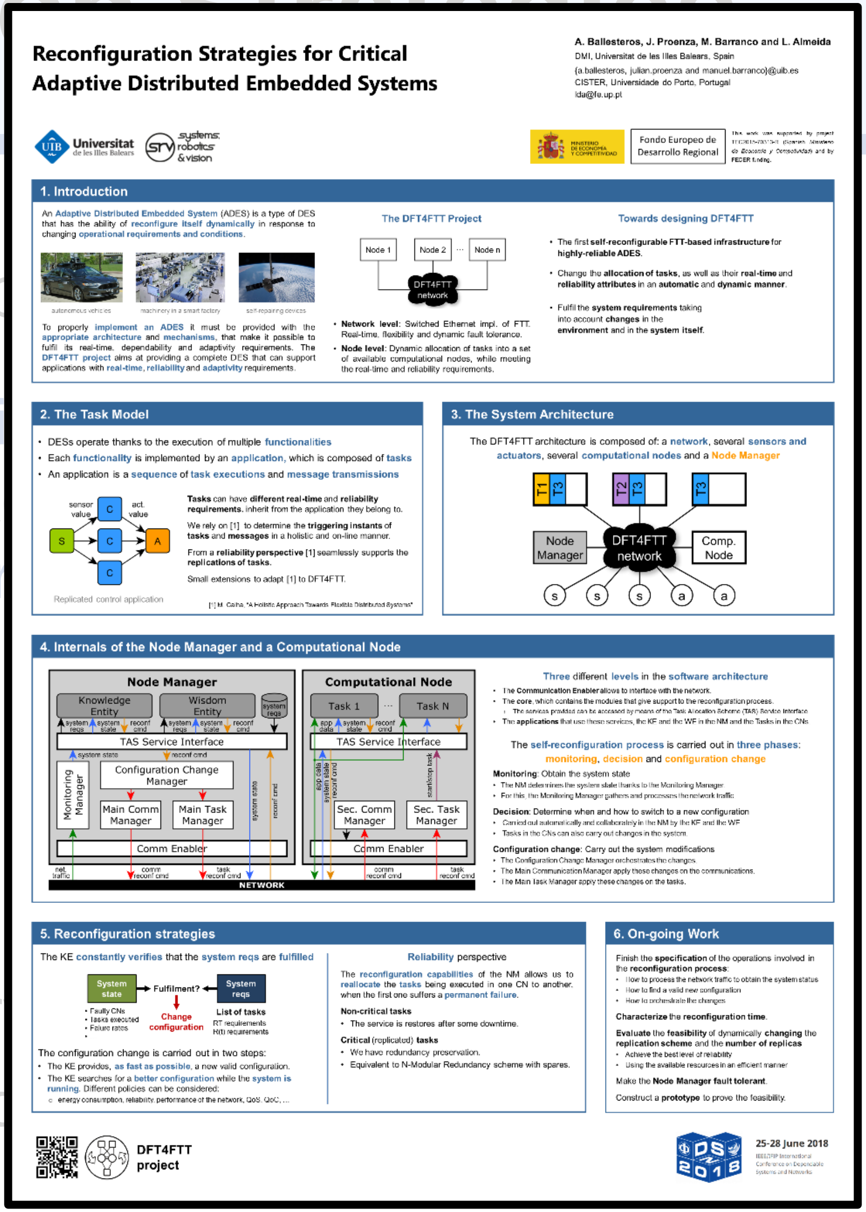
The reconfiguration capabili
reallocate the tasks being ex
when the first one suffers a per

Non-critical tasks

- The service is restores after

Critical (replicated) tasks

- We have redundancy preser
- Equivalent to N-Modular Red



Reconfiguration Strategies for Critical Adaptive Distributed Embedded Systems

A. Ballesteros, J. Proenza, M. Barranco and L. Almeida
DMI, Universitat de les Illes Balears, Spain
{a.ballesteros, julian.proenza and manuel.barranco}@uib.es
CISTER, Universidade do Porto, Portugal
lda@fe.up.pt

Universitat de les Illes Balears
systems, robotics & vision
MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
Fondo Europeo de Desarrollo Regional

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

1. Introduction

An **Adaptive Distributed Embedded System** (ADES) is a type of DES that has the ability of **reconfigure itself dynamically** in response to changing **operational requirements and conditions**.

autonomous vehicles

machinery in a smart factory

self-repairing devices

To properly **implement an ADES** it must be provided with the **appropriate architecture** and **mechanisms**, that make it possible to fulfil its real-time, dependability and adaptivity requirements. The **DFT4FTT project** aims at providing a complete DES that can support applications with **real-time**, **reliability** and **adaptivity** requirements.

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.

Towards designing DFT4FTT

- The first **self-reconfigurable FTT-based infrastructure** for **highly-reliable ADES**.
- Change the **allocation of tasks**, as well as their **real-time** and **reliability attributes** in an **automatic** and **dynamic manner**.
- Fulfil the **system requirements** taking into account **changes** in the **environment** and in the **system itself**.

2. The Task Model

- DESs operate thanks to the execution of multiple **functionalities**
- Each **functionality** is implemented by an **application**, which is composed of **tasks**
- An application is a **sequence** of **task executions** and **message transmissions**

Replicated control application

Tasks can have **different real-time** and **reliability requirements**. inherit from the application they belong to.

We rely on [1] to determine the **triggering instants** of **tasks** and **messages** in a holistic and on-line manner.

From a **reliability perspective** [1] seamlessly supports the **replications of tasks**.

Small extensions to adapt [1] to DFT4FTT.

[1] M. Calha, "A Holistic Approach Towards Flexible Distributed Systems"

3. The System Architecture

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

4. Internals of the Node Manager and a Computational Node

Three different **levels** in the **software architecture**

- The **Communication Enabler** allows to interface with the network.
- The **core**, which contains the modules that give support to the reconfiguration process.
  - The services provided can be accessed by means of the Task Allocation Scheme (TAS) Service Interface
- The **applications** that use these services, the KE and the WE in the NM and the Tasks in the CNs

The **self-reconfiguration process** is carried out in **three phases**: **monitoring**, **decision** and **configuration change**

Monitoring: Obtain the system state

- The NM determines the system state thanks to the Monitoring Manager.
- For this, the Monitoring Manager gathers and processes the network traffic

Decision: Determine when and how to switch to a new configuration

- Carried out automatically and collaborately in the NM by the KE and the WE
- Tasks in the CNs can also carry out changes in the system.

Configuration change: Carry out the system modifications

- The Configuration Change Manager orchestrates the changes.
- The Main Communication Manager apply these changes on the communications.
- The Main Task Manager apply these changes on the tasks.

5. Reconfiguration strategies

The KE **constantly verifies** that the **system reqs** are **fulfilled**

The configuration change is carried out in two steps:

- The KE provides, **as fast as possible**, a new valid configuration.
- The KE searches for a **better configuration** while the **system is running**. Different policies can be considered:
  - energy consumption, reliability, performance of the network, QoS, QoC, ...

Reliability perspective

The **reconfiguration capabilities** of the NM allows us to **reallocate** the **tasks** being executed in one CN to another, when the first one suffers a **permanent failure**.

Non-critical tasks

- The service is restores after some downtime.

Critical (replicated) **tasks**

- We have redundancy preservation.
- Equivalent to N-Modular Redundancy scheme with spares.

6. On-going Work

Finish the **specification** of the operations involved in the **reconfiguration process**:

- How to process the network traffic to obtain the system status
- How to find a valid new configuration
- How to orchestrate the changes

Characterize the **reconfiguration time**.

Evaluate the **feasibility** of dynamically **changing** the **replication scheme** and the **number of replicas**

- Achieve the best level of reliability
- Using the available resources in an efficient manner

Make the **Node Manager fault tolerant**.

Construct a **prototype** to prove the feasibility.

DFT4FTT project

25-28 June 2018
IEEE/IFIP International Conference on Dependable Systems and Networks