RÉPUBLIQUE FRANÇAISE
*Liberté*
*Égalité*
*Fraternité*

cnes
CENTRE NATIONAL D'ÉTUDES SPATIALES

# TUNING LINUX FOR EMBEDDED SPACE APPLICATIONS

**Simon CORBIN**

**Software engineer**

**CNES, France**

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

25-27 August 2025
Amsterdam,
The Netherlands

# WHAT IS CNES ?

© NASA, ESA, CSA/M. McCaughrean, S. Pearson, 2023

# WHAT IS CNES ?

**French space agency**

**Founded in 1961**

**+2400 employees**

Maintaining and reinforcing the strategic autonomy of France and Europe

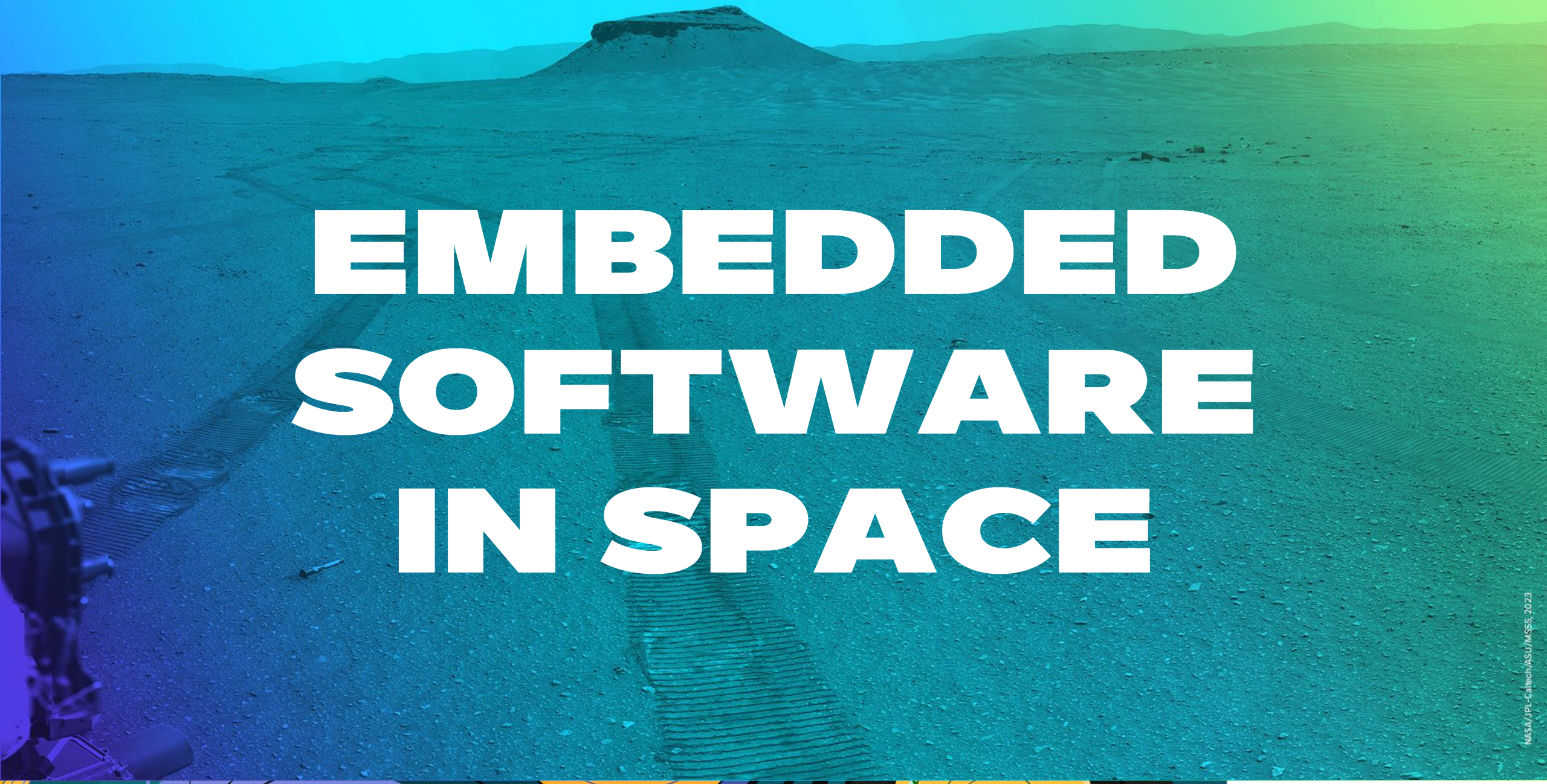
Supporting and promoting scientific excellence

Supporting the competitiveness of the space ecosystem

Committing to a sustainable world

For more information : https://cnes.fr

# EMBEDDED SOFTWARE IN SPACE



NASA/JPL-Caltech/ASU/MSSS 2023

# TYPICAL SPACECRAFT ARCHITECTURE



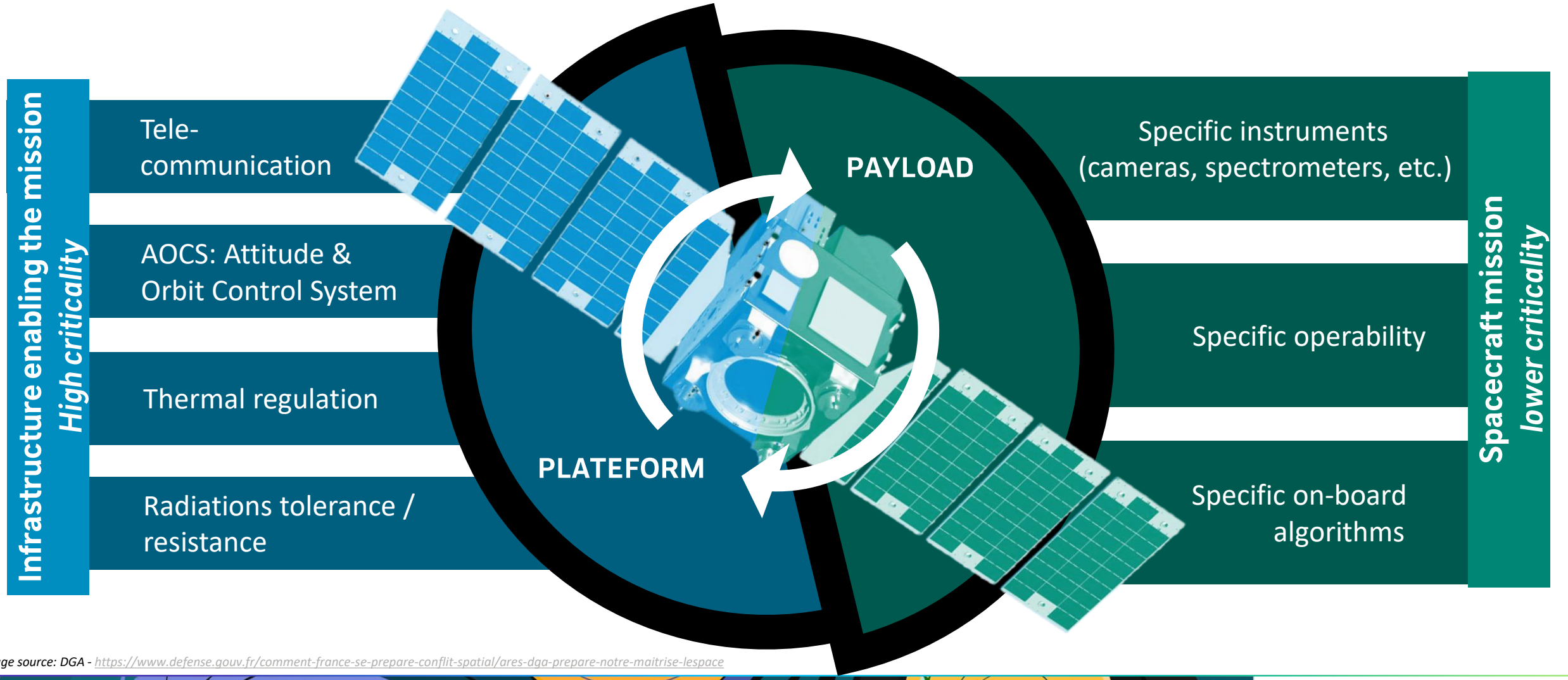
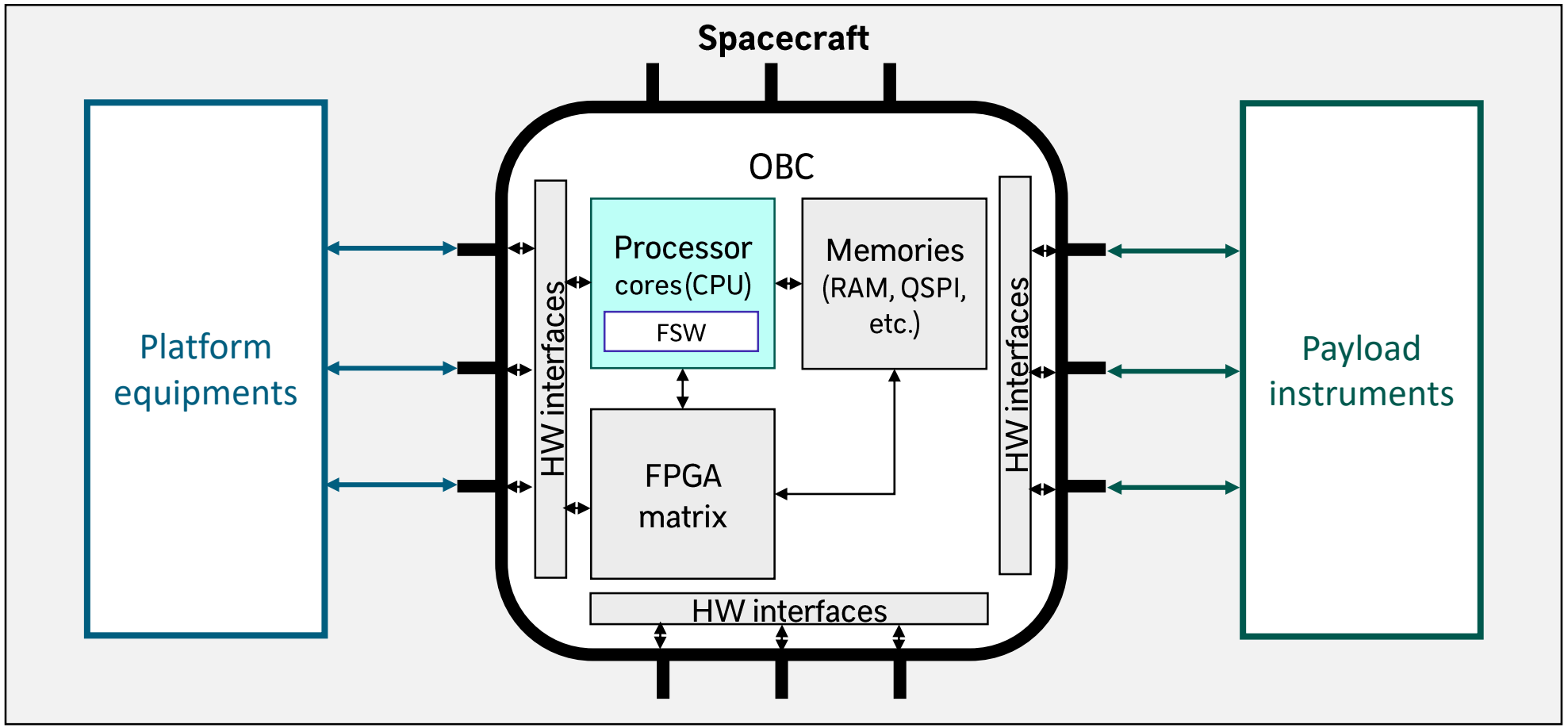



*Image source: DGA - https://www.defense.gouv.fr/comment-france-se-prepare-conflit-spatial/ares-dga-prepare-notre-maitrise-lespace*

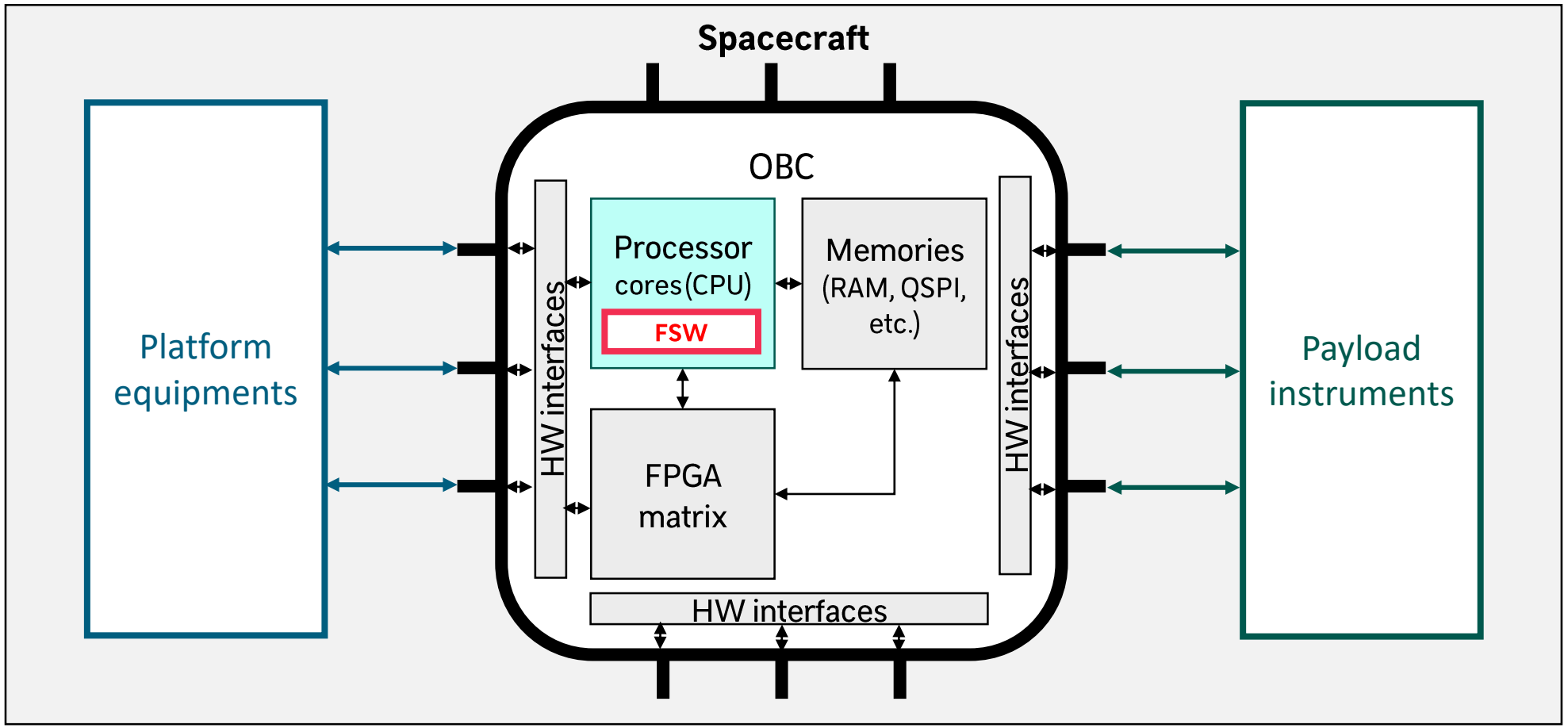
# ON-BOARD COMPUTER





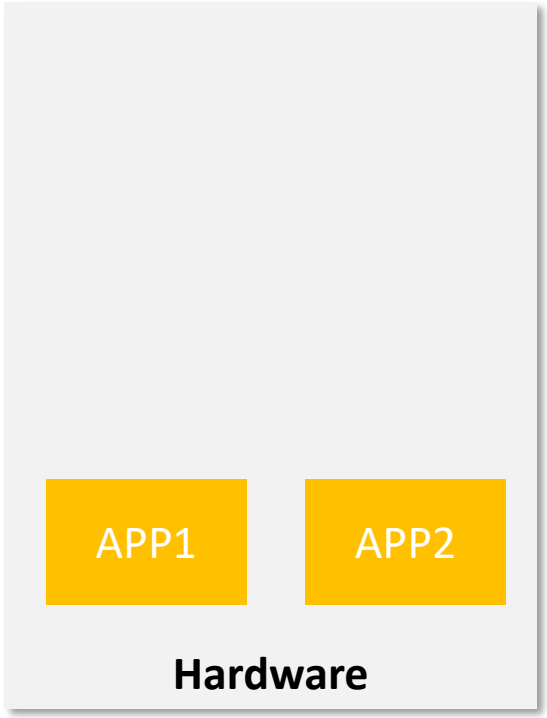
# ON-BOARD COMPUTER



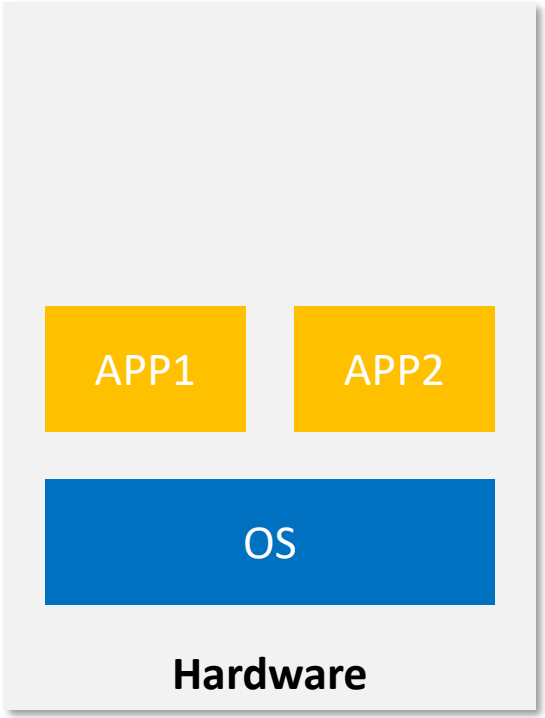


# TYPICAL ON-BOARD SOFTWARE ARCHITECTURES

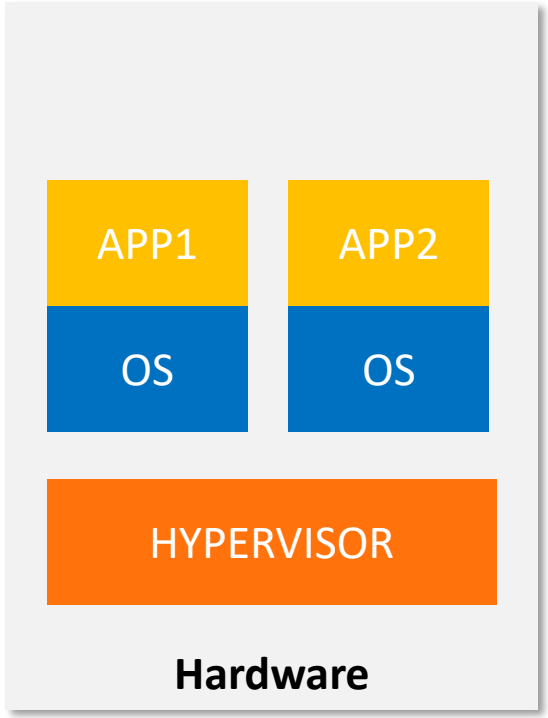





- No operating system
- Simple applications
- Low complexity level
- Small payloads/instruments



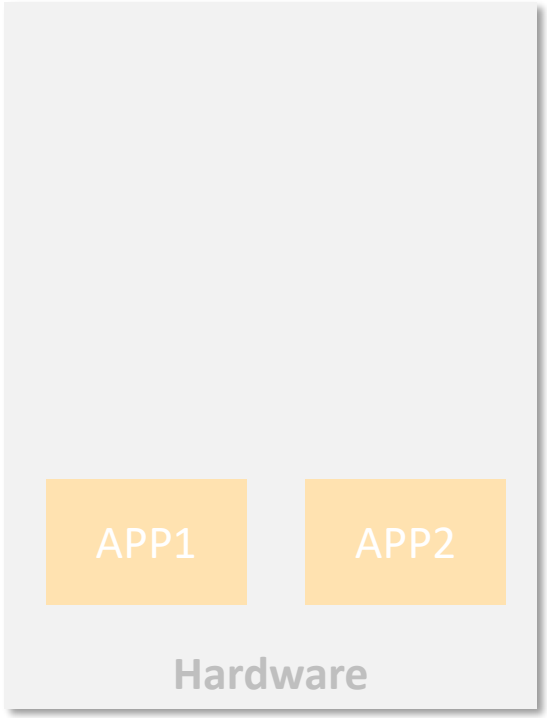

- OS needed for tasks scheduling
- Platforms and payloads



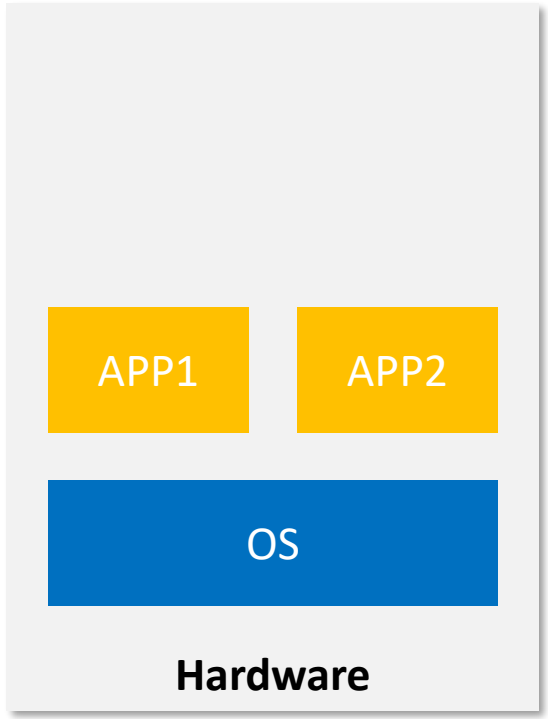

- Time and Space partitioning
- Mixed criticality
- Qualified Hypervisor (ECSS B)

# TYPICAL ON-BOARD SOFTWARE ARCHITECTURES

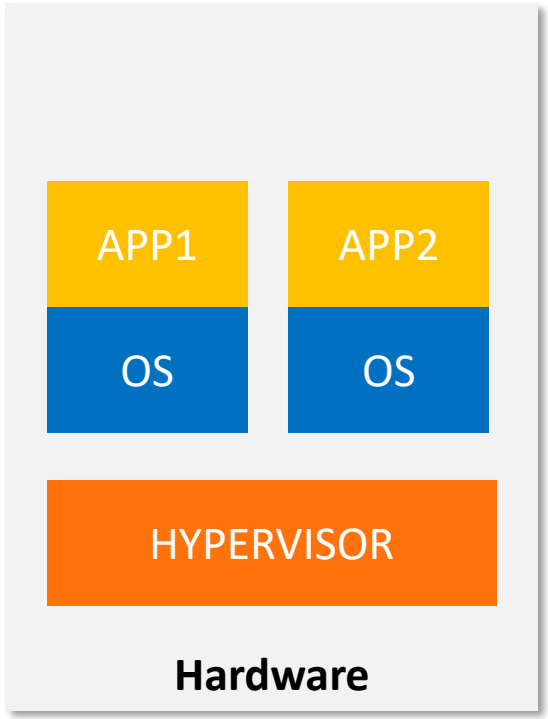





- No operating system
- Simple applications
- Low complexity level
- Small payloads/instruments



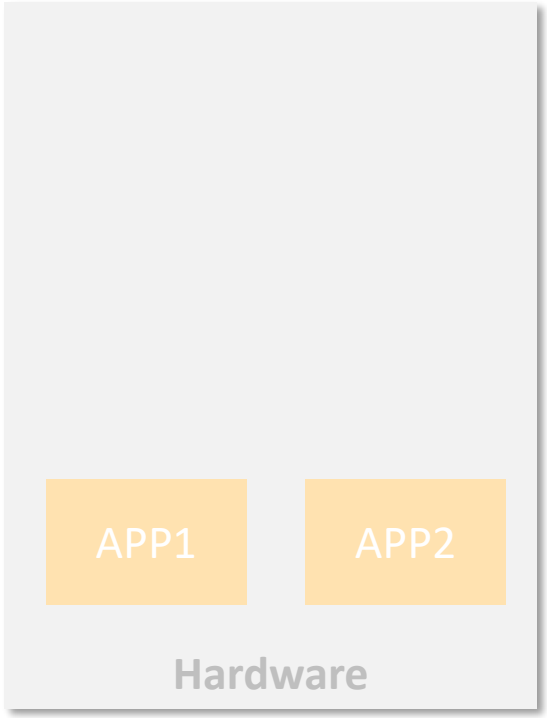

- OS needed for tasks scheduling
- Platforms and payloads



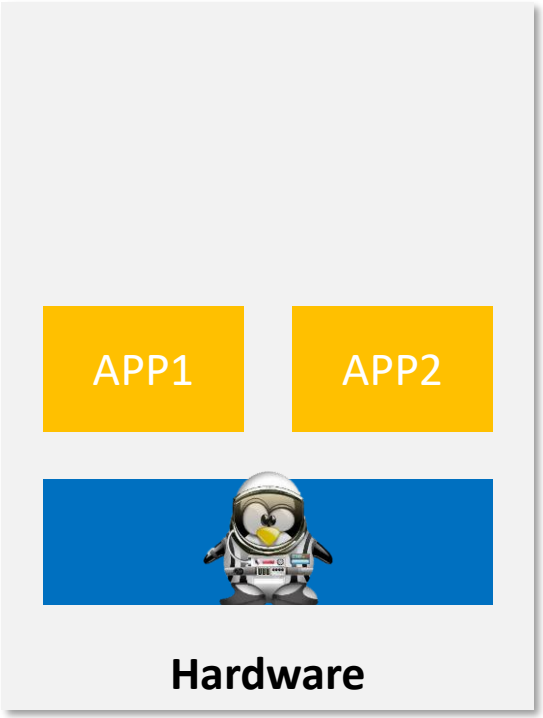

- Time and Space partitioning
- Mixed criticality
- Qualified Hypervisor (ECSS B)

# TYPICAL ON-BOARD SOFTWARE ARCHITECTURES

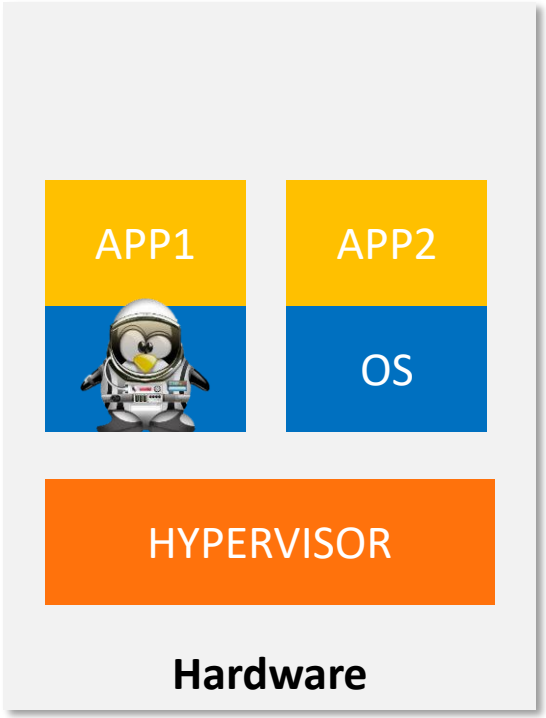





- No operating system
- Simple applications
- Low complexity level
- Small payloads/instruments




- OS needed for tasks scheduling
- Platforms and payloads




- Time and Space partitioning
- Mixed criticality
- Qualified Hypervisor (ECSS B)

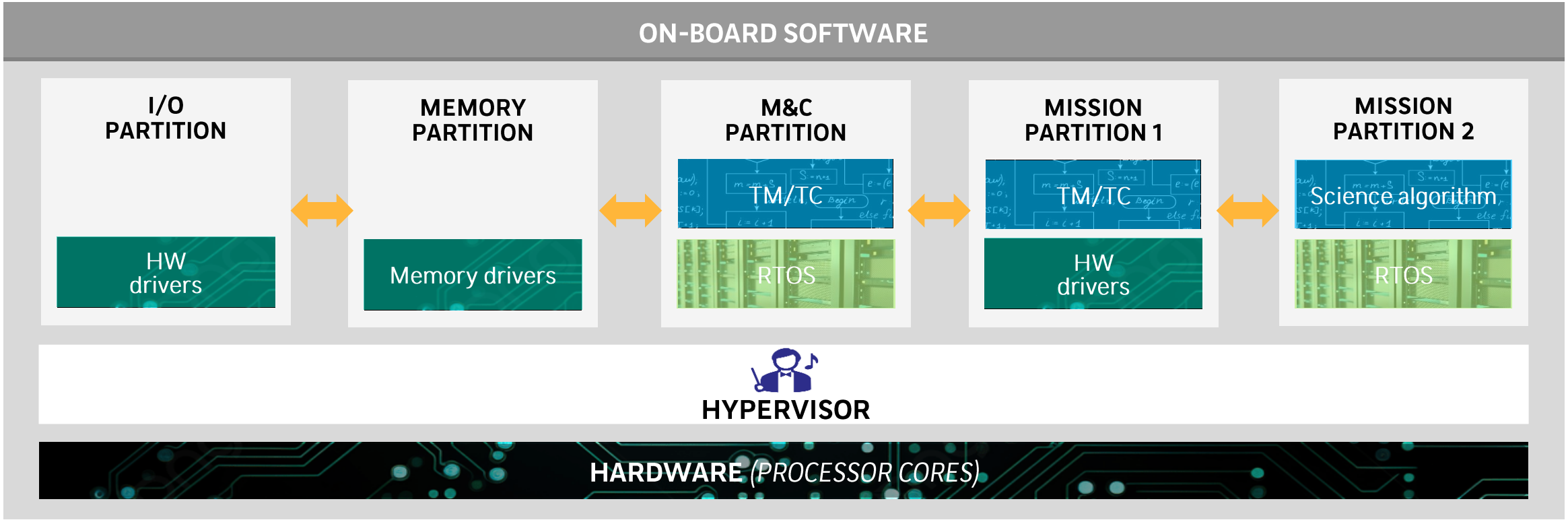
# FLIGHT SOFTWARE EXAMPLE



## TSP: an architectural concept developed for the aeronautic industry

- implemented by Boeing and Airbus in the frame of the new aircraft projects (Airbus A380)

**Allows strict separation of different applications on the same computer**

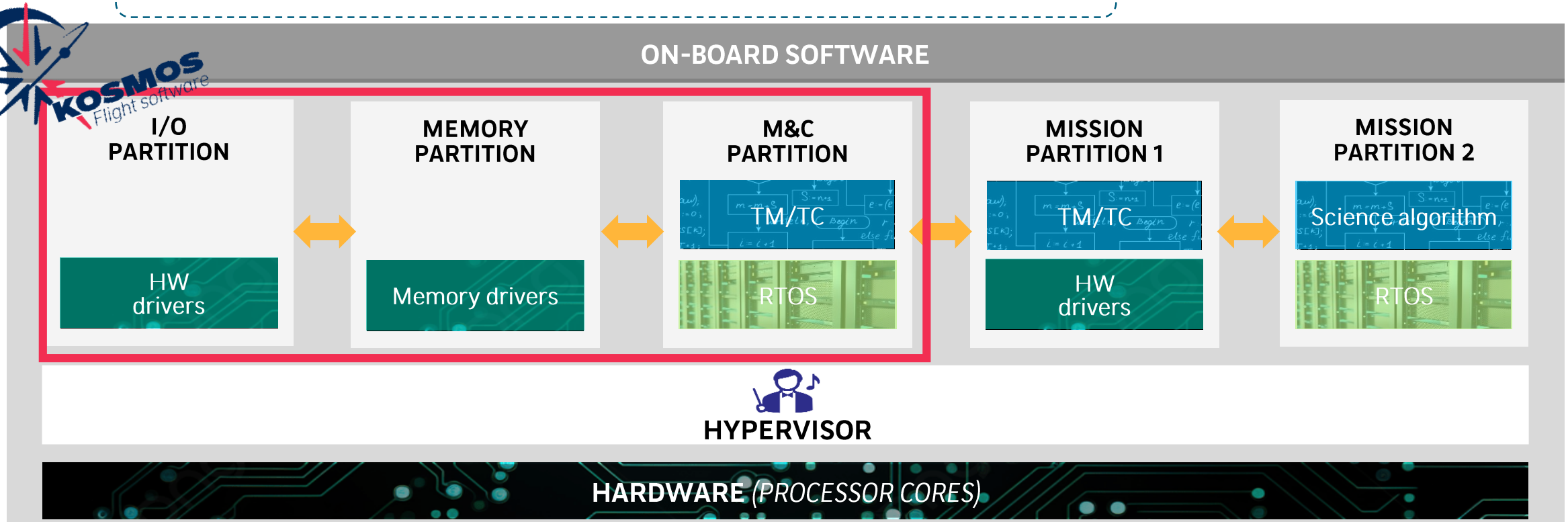


# FLIGHT SOFTWARE EXAMPLE

## TSP: an architectural concept developed for the aeronautic industry

- implemented by Boeing and Airbus in the frame of the new aircraft projects (Airbus A380)

**Allows strict separation of different applications on the same computer**

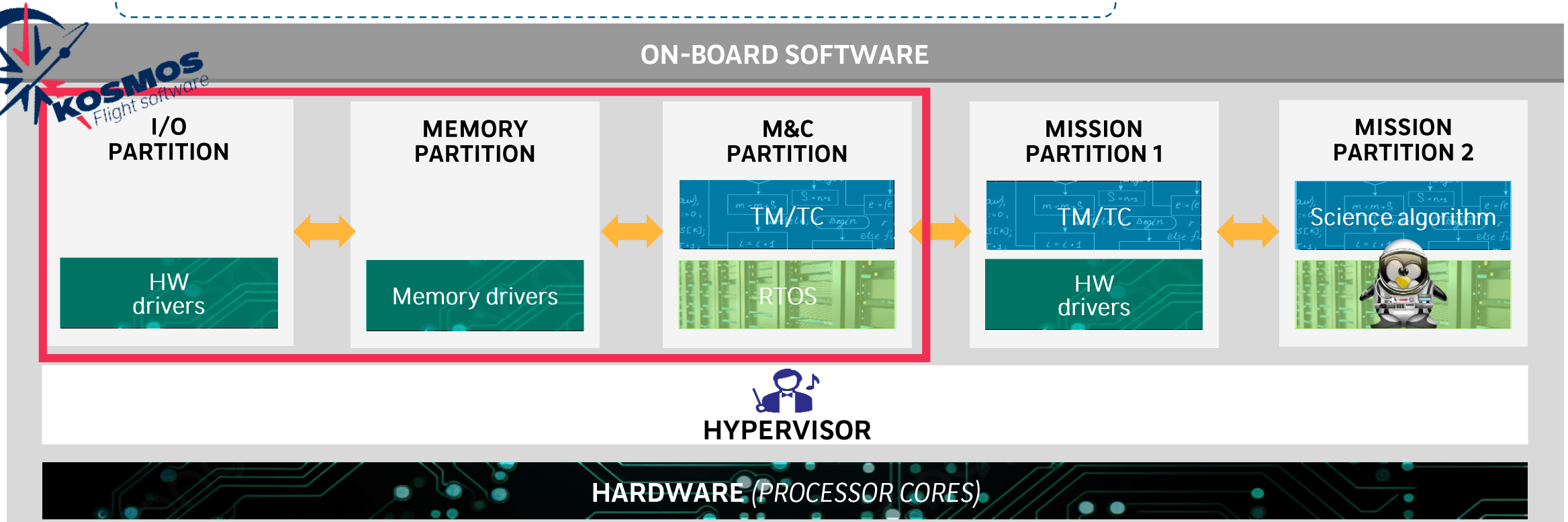


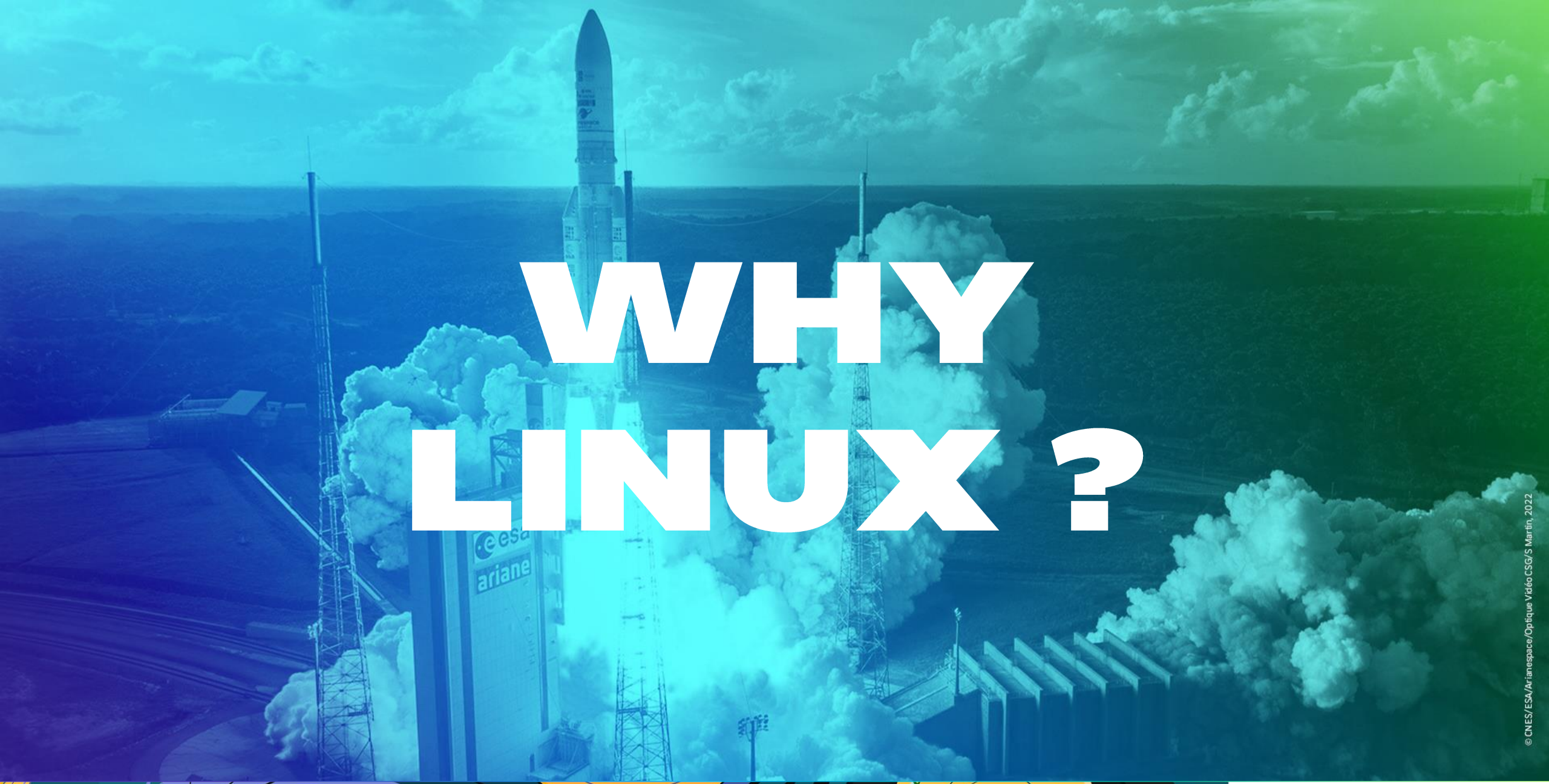
# FLIGHT SOFTWARE EXAMPLE

## TSP: an architectural concept developed for the aeronautic industry

- implemented by Boeing and Airbus in the frame of the new aircraft projects (Airbus A380)

**Allows strict separation of different applications on the same computer**






# WHY LINUX ?

© CNES/ESA/Arianespace/Optique Vidéo CSG/S Martin, 2022

# ON-BOARD SOFWARE CHALLENGES

## REDUCE TIME-TO-MARKET OF RELIABLE FLIGHT SW

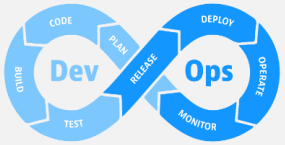
### Maximize reuse of SW building blocks

### Generic modular architectures

KOSMOS Flight software

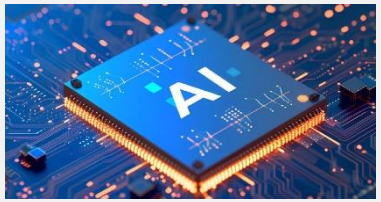
yocto PROJECT

### Adapt SW engineering processes




## INCREASE AUTONOMY AND ON-BOARD INTELLIGENCE

### On-board AI




### Autocoding

### Complex algorithms

Development and validation strategy has to adapt to this increase of complexity

# ON-BOARD SOFTWARE CURRENT STATUS

## Today

- Most applications coded in C language
- Specific hardware
(radiation tolerant for example)
- Specific software
(compliant to space standards)
- Small community
- Lack of support
- Increasing complexity (both HW and SW)

NEW
SPACE

## Tomorrow with Linux ?

- Allow the use of high level languages
- Ease application porting
- Plenty of compatible targets
- Large community
- Lot of support / documentation

***A magic solution… really ?***

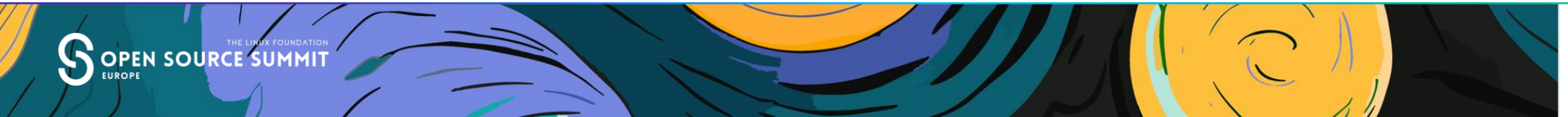
# USING LINUX : WHAT DO WE FEAR ?

2024 : collaboration with major space actors

- Elaborate a non-exhaustive list about Linux feared events (for use in space)
- Propose recommendations/solutions for each element of the list

Main topics :

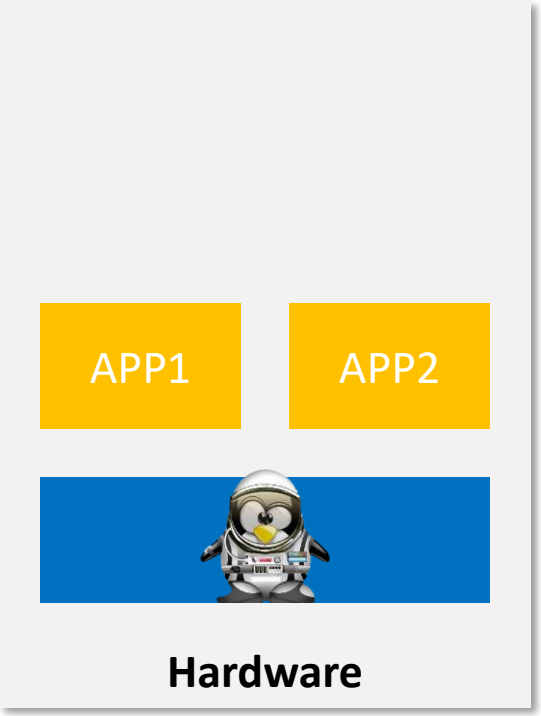
- Real time constraints (memory, timers, scheduling…)
- Security (updates, unsafe languages, kernel complexity, ground/on-board communication…)
- Base level system requirements (kernel configuration, service management, system initialization, log system, file system…)
- Deployment / OTA updates
- Licence / export control



# ONGOING WORK

© CNES/DERAMECOURT Arnaud, 2007

# LEVEL 1 : ONE PAYLOAD APPLICATION ON A DEDICATED CPU

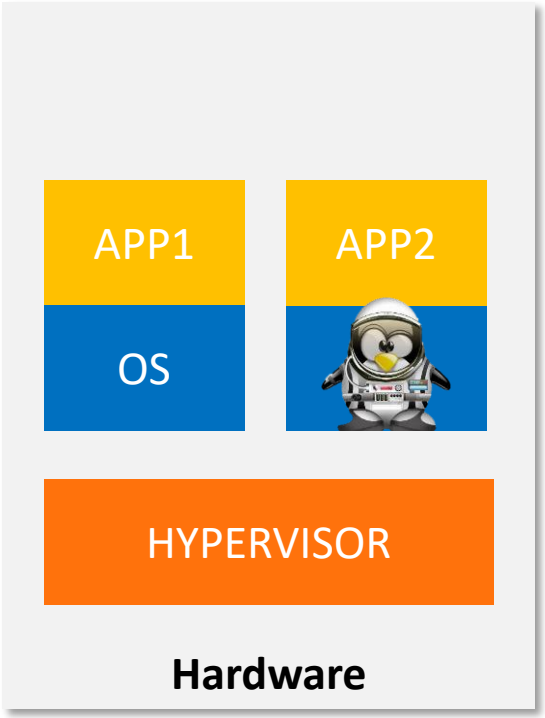



- Minimize risk of disrupting critical applications
- Higher cost
- Higher consumption
- Takes more space
- Already proven solution

➢ Low interest : Won’t fly in the short term

# LEVEL 2 : LINUX PARTITION RUNNING NEXT TO CRITICAL ONES ON THE SAME CPU

ONGOING WORK

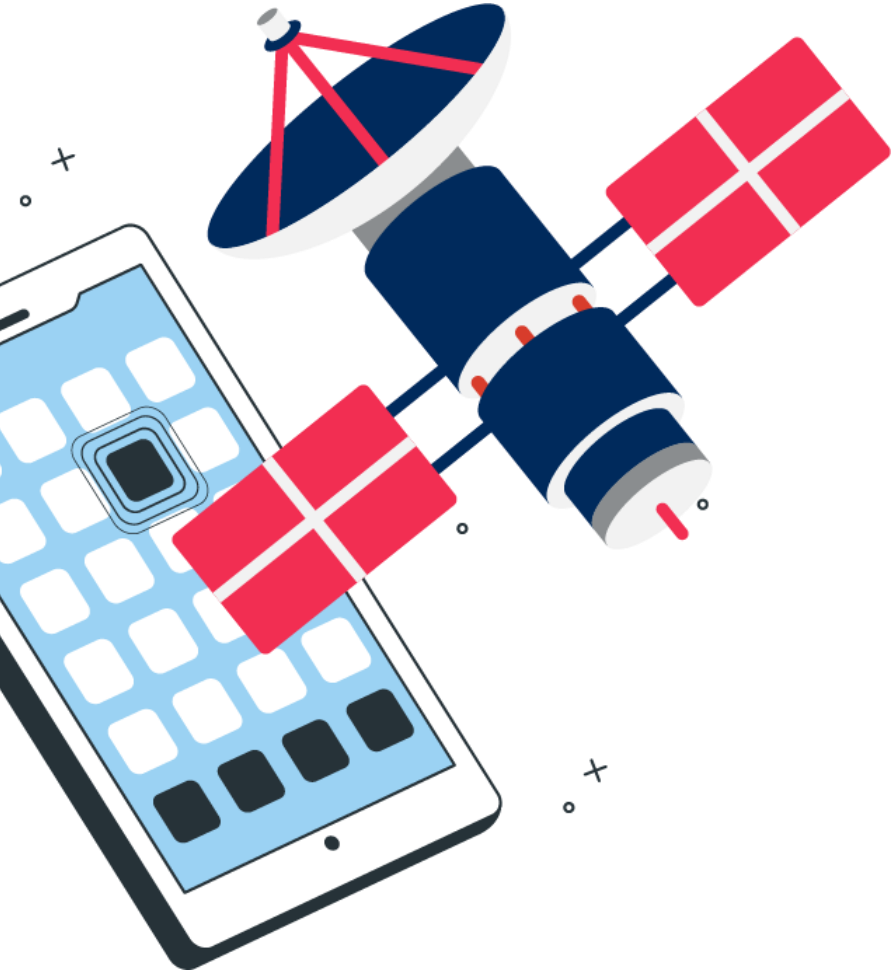



- Both critical/non-critical applications on the same CPU
- Lower cost
- Lower consumption
- Only one payload application running on Linux

➢ Will fly on AEROSAT, a nano-satellite developed by students at CNES within the NANOLAB academy

# LEVEL 3 : ON-BOARD « APP STORE » RUNNING ON LINUX





- Embedding an application catalog
- Switch to another application when the experience is done
- Load new applications from ground
- Delete on-board applications if they are not needed anymore

➢ Nothing less than a smartphone in orbit !

# WHAT'S ABOUT THE DISTRO ?

© CNES/ESA/Arianespace/Optique Vidéo CSG/S Martin, 2022

# SOME FEATURES ABOUT THE DISTRO



**User App layer**

**PUS layer** (communication protocol for telecomands/telemetry)

**meta-layer-xng (hypervisor)**

**Yocto specific layer meta (meta-yocto)**

- Starts from poky (core-image-minimal)
- Scarthgap branch
- Use of PREMPT-RT patch
- Use of systemd for service management
- Following recommendations from the activity with space actors (see slide 17)
- Use of specific XNG (hypervisor) layer (Linux as guestOS + inter partitions communication)
- Use of specific
- Only one payload application running on Linux

# THANK YOU !

© CNES/DERAMECOURT Arnaud, 2007