

Arquitectura del sistema de gestión de datos

Objetivos

- Conocer el propósito, importancia y principios básicos de las tareas de captura de requisitos y análisis de sistemas computacionales.
- Conocer principios generales de modelado y especificación de software y hardware.
- Comprender la importancia de un diseño adecuado de la arquitectura del sistema: hardware y software.

Índice

- Objetivos
- Bibliografía recomendada
- Introducción
- Requisitos funcionales y no funcionales
- Arquitectura de hardware
- Arquitectura de software
- Caso de uso UPMSat-2

Bibliografía

- Sommerville, Ian. “Software Engineering”, Pearson, 2016, 10th Edition.
- Grady Booch, James Rumbaugh, Ivar Jacobson. “The Unified Modeling Language User Guide” Addison-Wesley Publishing Co. 1999, 2nd Edition.
- Booch, G and Bryan, D. “Software Engineering with Ada”, Addison-Wesley Publishing Co. 1994 3rd Edition.

Introducción

- Los requisitos de un sistema o subsistema son descripciones de servicios que debe proporcionar y sus restricciones de operación.
- Un requisito es una capacidad, característica, condición o factor de calidad que un sistema o subsistema debe proporcionar para tener utilidad para sus usuarios.
- El proceso de encontrar, documentar, analizar, organizar y monitorizar los requisitos (generalmente cambiantes) de un sistema se llama ingeniería de requisitos.

Requisitos

- Hay distintos niveles de requisitos según el grado especificación.
 - Requisitos de alto nivel: un servicio que debe proporcionar el sistema enunciado de forma muy abstracta.
 - Requisitos de bajo nivel: una descripción detallada de una funcionalidad del sistema en la que incluso se pueden definir los parámetros formales.
- Los requisitos de usuario son de alto nivel y a partir de ellos hay que extraer requisitos más detallados.
 - Se suelen recoger en un documento para que distintas compañías o consorcios hagan sus ofertas.

Ejemplo: ESA Statement of Work

A2.1. Functional and Performance Requirements

##ETSPRTB-SOW-FR-10

The CSW shall be compatible with LEON3-FT and NGMP processors equipped with MMU.

##END-REQ

##ETSPRTB-SOW-FR-20

The CSW V6 shall use a separation uKernel to manage the SW partitions.

##END-REQ

##ETSPRTB-SOW-FR-30

The CSW V6 shall support single-core and multi-core configurations.

##END-REQ

##ETSPRTB-SOW-FR-40

The CSW V6 shall exploit the multi-core capabilities of the NGMP board.

##END-REQ

##ETSPRTB-SOW-FR-50

The SVF shall be compatible with LEON3-FT and NGMP processors.

##END-REQ

##ETSPRTB-SOW-FR-60

The CSW shall maintain the following SW partitions:

- AOCS partition
- DMS partition
- I/O partition
- GoldenEye payload partition
- FDIR partition

##END-REQ

Captura de requisitos

- Los requisitos de un sistema se suelen obtener de forma iterativa.
- Existen herramientas que ayudan en este proceso de ingeniería de requisitos.
- Lo fundamental es el análisis y la comprensión del sistema que se necesita y se va a construir.
- La lista de requisitos de un sistema se recoge en la especificación de requisitos del sistema.
 - Corrección
 - Completitud
 - Consistencia

Clasificación de requisitos

Los requisitos del software se clasifican tradicionalmente en:

- **Funcionales:** Funciones que describen las funciones del sistema. Es decir, las relaciones entre las entradas y las salidas.
- **No funcionales o extra-funcionales:**
 - Definen las propiedades de un sistema que no están relacionadas con su funcionalidad.
 - Uso de memoria, prestaciones, uso de estándares, ...
 - Generalmente se aplican a nivel de sistema y no a nivel de función.
 - La mantenibilidad es una propiedad del sistema y no de una función específica pero condiciona cómo se debe desarrollar el software.

Requisitos no funcionales

ECSS-Q-ST-80 mentions the following NFR

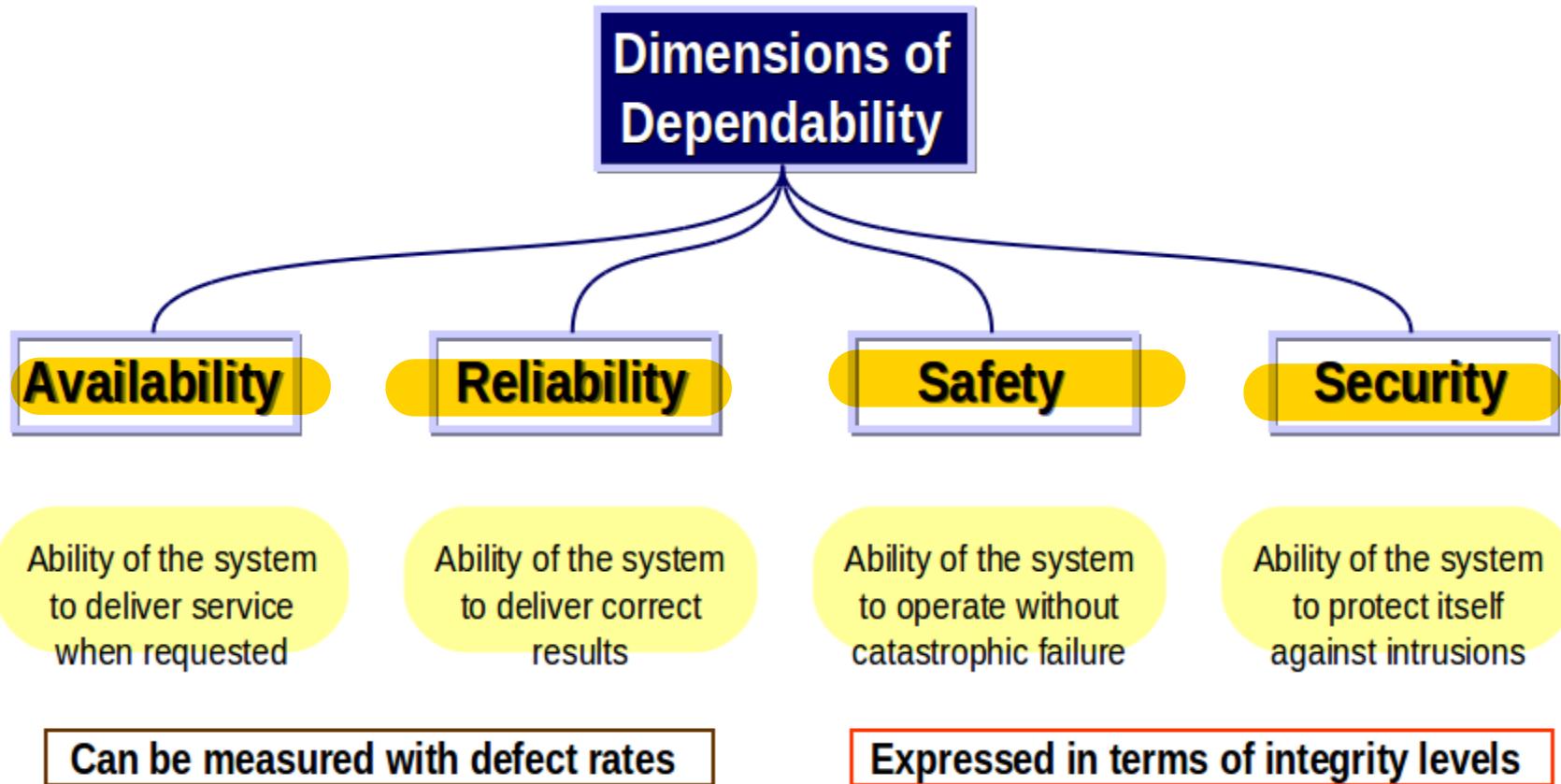
- performance,
 - including real-time and numerical accuracy
- safety,
- reliability,
- robustness,
- quality,
- maintainability,
- configuration management,
- security,
- privacy,
- metrification,
- verification and validation.

Examples of real-time requirements

- Absolute time execution
 - e.g. the horizon sensor must be calibrated at 1200 UT
not very common
- Periodic execution
 - e.g. the launcher trajectory must be adjusted every 100 ms
- Sporadic execution with minimal separation
 - e.g. telecommands may arrive at any time with a minimal inter-arrival time of 1500 ms
- Deadline
 - e.g. telemetry messages must be processed within 500 ms of their arrival

Dependability

- Dependability: the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers (IFIP WP 10.4).



Requisitos legales del software

- Violación de patentes y derechos de autor
 - Hay que distinguir entre licencias y propiedad
- Seguridad y leyes de privacidad
 - Si se custodia información privada
- Instrucciones de seguridad (safety)
 - Herramientas o utensilios peligrosos que contienen software

UPMSat-2

Technical Specification.

Software Requirements

Specification

Análisis de requisitos

- Documento SSS — Software System Specification
 - Funcionalidad
 - Restricciones
 - Entorno de funcionamiento
 - Carga útil
 - Requisitos
 - Verificación, validación e integración del sistema
- Los requisitos deben ser
 - Completos
 - Consistentes
 - Correctos

Funciones del software embarcado

- Adquisición y procesado de telemetría (TM).
- Descodificación y ejecución de órdenes remotas (TC)
- Supervisión y control de la plataforma
- Control y determinación de actitud (ADCS)
- Adquisición de datos de mantenimiento (housekeeping)
- Detección y tratamiento de fallos
- Adquisición, gestión y almacenamiento de datos de la carga útil
- Registro de datos temporales

Funciones del software de tierra

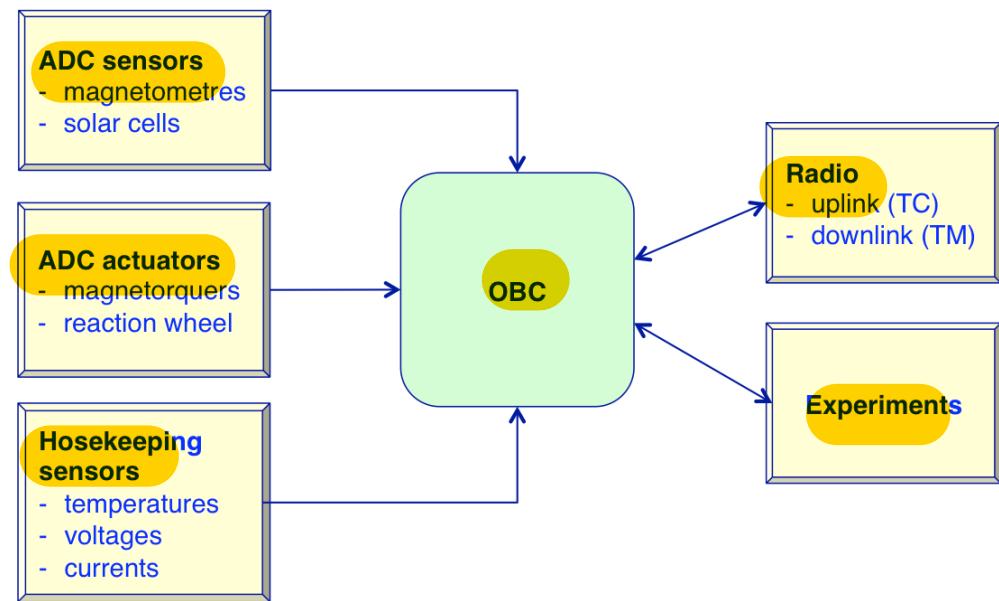
- Cálculos de parámetros orbitales y tiempos de paso
- Determinación de la posición del satélite
- Recepción, procesamiento y registro de datos de telemetría (TM)
- Entrada de órdenes de operador y envío al satélite mediante mensajes de órdenes remotas (TC)
- Gestión de la interfaz de operador

Restricciones

- Uso de estándares
 - ECSS-E-ST-40C y ECSS-Q-ST-80C
 - Unidades SI
- Software libre
 - GPL / LGPL
- Lenguaje de alto nivel fiable
 - Ada /SPARK

Entorno y carga útil

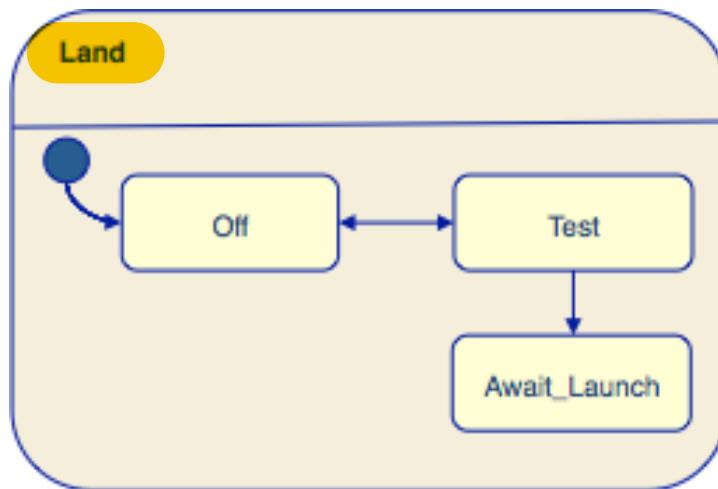
- Entorno de funcionamiento
 - Características del satélite
 - Características del OBC
 - Diagrama de contexto
 - Interfaces de hardware
- Carga útil
 - experimentos



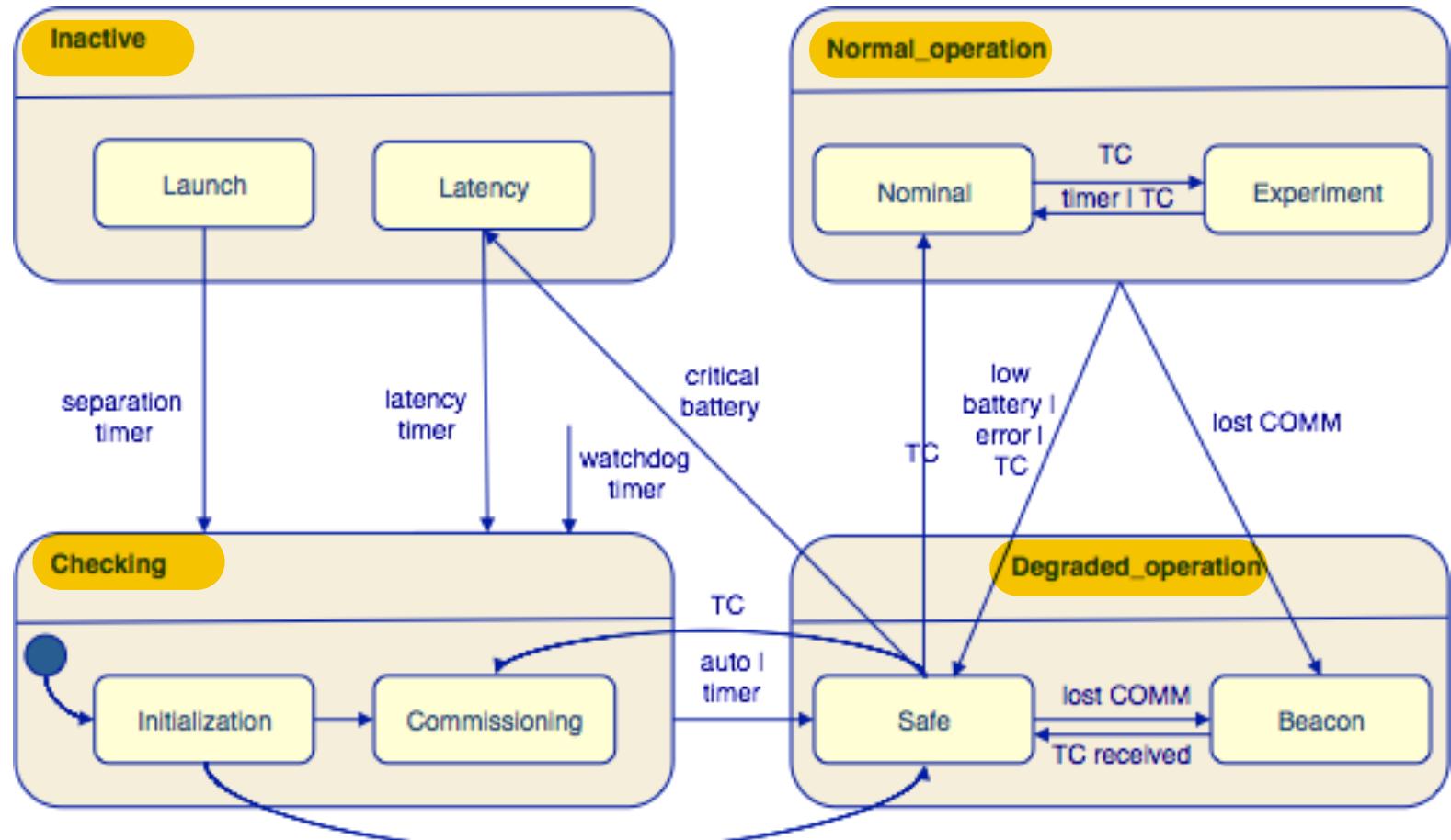
Requisitos específicos

- Funcionamiento del sistema
 - modos de funcionamiento
 - órdenes remotas (TC)
 - telemetría (TM)
- Control de actitud (ADCS)
- Supervisión de la plataforma (*housekeeping*)
- Almacenamiento y registro de datos
- Interfaces
- Computador
- Seguridad, fiabilidad, disponibilidad
- Diseño y herramientas de software
- Mantenimiento del software

Modos de funcionamiento en tierra



Modos de funcionamiento en vuelo



Órdenes remotas (TC)

Requisitos	Descripción
Configure	Modificación de los parámetros de configuración del software.
Commission	Pasa a modo de puesta en servicio.
OpenLink	Inicio de comunicación. Envía a tierra los datos: <ul style="list-style-type: none">- Estado de la plataforma (<i>housekeeping</i>)- Registro (logbook).- Estado de magnetópares (<i>health table</i>)- Reloj de misión (<i>mission clock</i>)
Nominal	Pasa a modo nominal.
Safe	Pasa a modo seguro.
Latency	Pasa a modo de latencia.
Logbook	Envía a tierra del contenido del registro.
Test	Verificación para ensayos en tierra.

Medidas periódicas (*housekeeping*)

Requisitos	Descripción
Temperaturas	<ul style="list-style-type: none">- Caras del satélite- Baterías- Magnetómetros- Computador- Experimentos
Actitud	Medida magnetómetro Salida magnetopares Intensidad magnetopares
Energía	Tensión en bus Tensión en batería Tensión en paneles solares Intensidad en batería Intensidad en paneles solares
Períodos	Parámetros de configuración

Telemetría (TM)

Requisitos	Descripción
TM periódica	Datos de sistema (<i>housekeeping</i>) Períodos: parámetros de configuración
TM de sucesos	Cambios de modo Cambios de estado de baterías Otros
TM de petición	Datos de sistema (<i>housekeeping</i>) Registro del sistema Tiempo de misión

Telemetría en distintas situaciones

Requisitos	Descripción
TM sin cobertura	Almacenar mensajes en registro
TM con cobertura	Los nuevas mensajes se envían cuando se generan El registro de mensajes se envía a tierra a petición Se distinguirán ambos tipos de mensajes
TM en funcionamiento degradado	Modo seguro: sólo mensajes básicos
TM en comprobación	Modo inicio o puesta en servicio: sólo mensajes iniciales
TM en radiofaro	Modo radiofaro: sólo mensaje de aviso periódico

Control de actitud

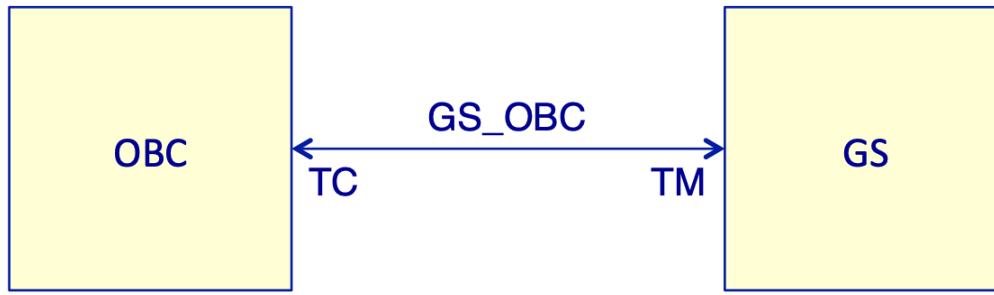
Requisitos	Descripción
Funciones	<ul style="list-style-type: none">- Lectura de magnetómetros- Cálculo de acción en magnetopares (algoritmo de control)- Emisión de señales de magnetopares
TC de ADCS	<ul style="list-style-type: none">- Sincronización del tiempo- Recepción de la posición del satélite- Órdenes para los sensores y actuadores- Configuración de magnetómetro- Configuración de sensores solares- Configuración de magnetopares- Actualización de efemérides- Actualización de parámetros de configuración del ADCS- Lectura de parámetros de configuración del ADCS
TM de ADCS	<ul style="list-style-type: none">- Envío de medidas de los sensores- Datos orbitales: actitud y posición orbital del satélite- Datos de estado del ADCS: campo magnético, posición solar, etc.- Parámetros de configuración del ADCS
Períodos	<ul style="list-style-type: none">- Lectura de magnetómetros- Control y accionamiento de magnetopares

Computador embarcado

Requisitos	Descripción
Condiciones de arranque	<ul style="list-style-type: none">- Apagado: al recibir alimentación- Encendido: señal <i>reset</i> en hardware
Acciones de arranque	<ul style="list-style-type: none">- Copia de código desde EEPROM a RAM- Comprobación de estado de los dispositivos- Ejecución en modo inicio
Temporizador de separación	<ul style="list-style-type: none">- Realizado en hardware- Se activa en el momento de la separación- Al vencer el tiempo activa la alimentación del computador
Temporizador de latencia	<ul style="list-style-type: none">- Realizado en hardware- Se activa al pasar al estado de latencia (batería baja)- Al vencer el tiempo activa la alimentación del computador
Temporizador de guardia (<i>watchdog timer</i>)	<ul style="list-style-type: none">- Realizado en hardware- Se activa periódicamente desde el software- Si llega a cero hace un <i>reset</i> del computador
Reloj de misión	Mide el tiempo transcurrido desde la separación (hardware)

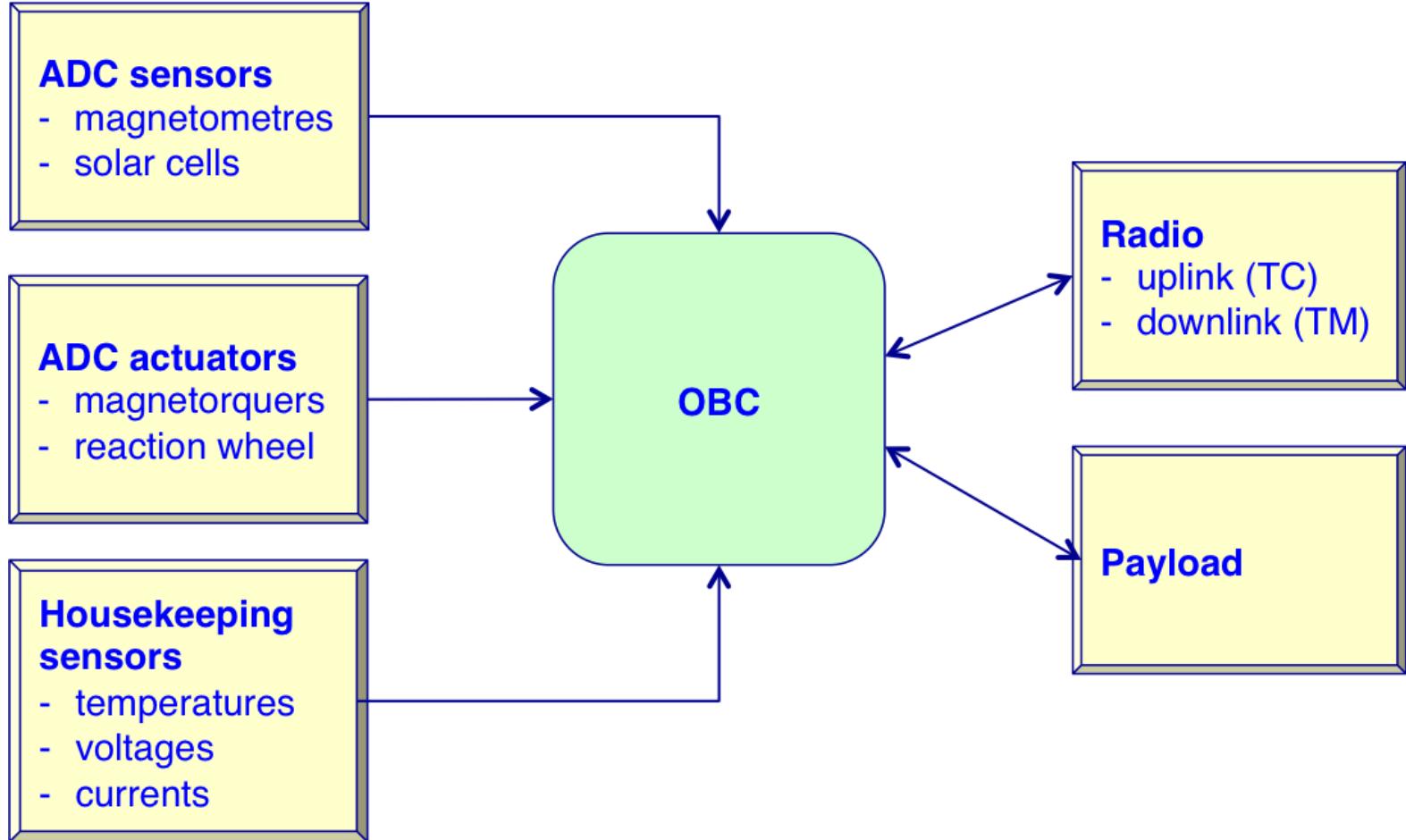
UPMSat-2 on-board software

Top-level architecture



- Data links
 - UHF U band(AMSAT): 435.00–438.00MHz
 - AUL: Uplink (telecommands): max 1200 bps
 - ADL: Downlink (housekeeping data): max 1200 bps
 - UHF space research band: 400.15 – 401.0MHz
 - RDL: Downlink (experiment data): max 9600 bps

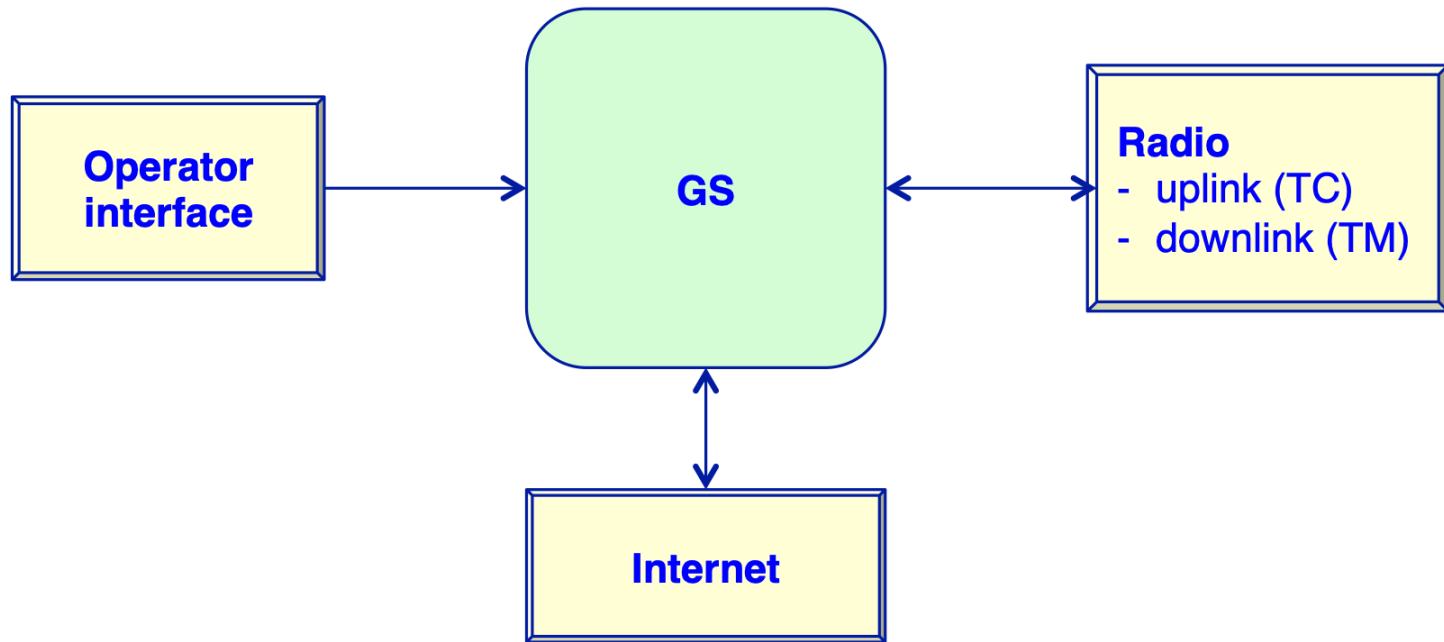
OBC context



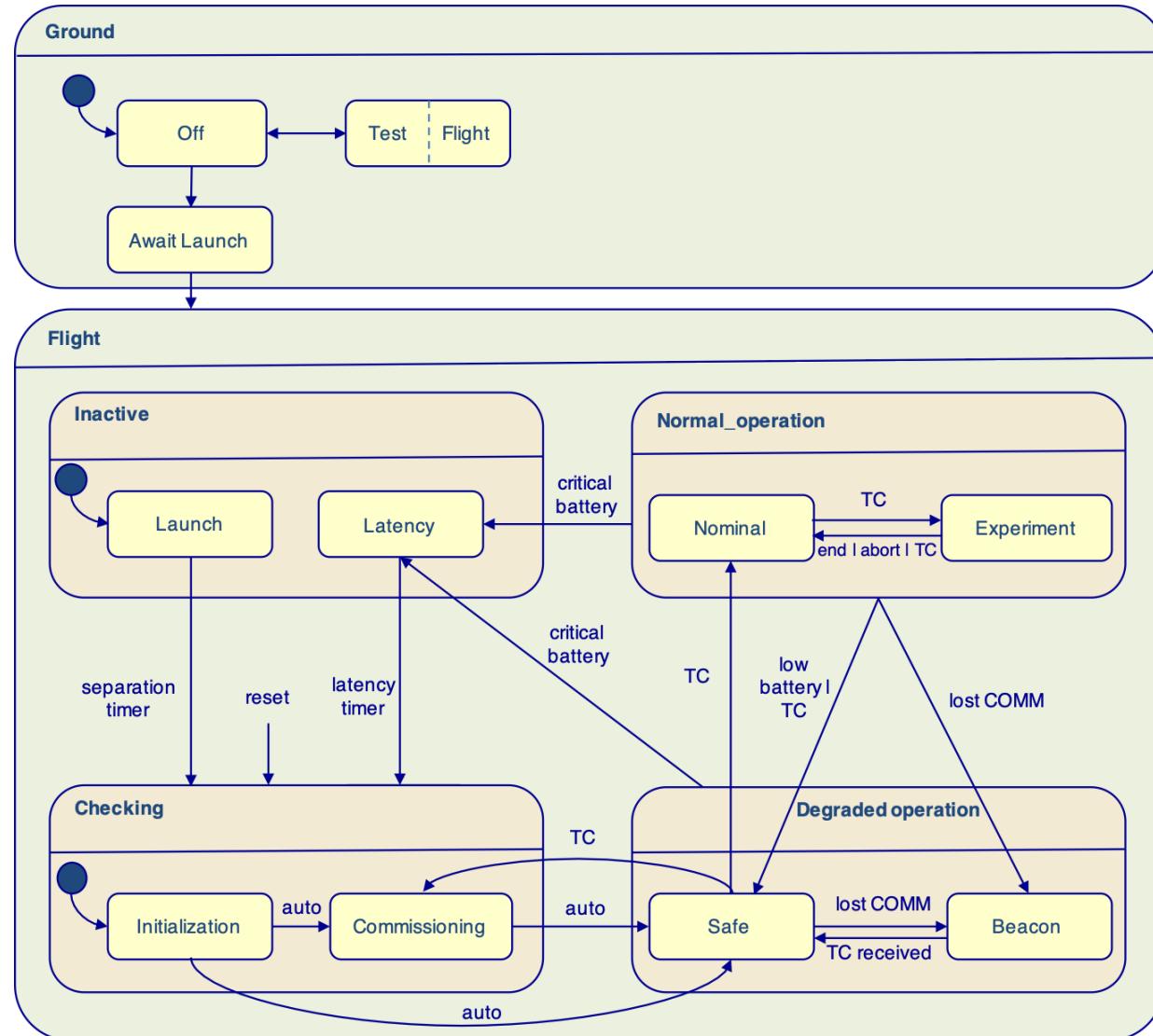
Experiment payload

- MTS Micro Thermal Switch (*IBERESPACIO*)
- RW Reaction Wheel (*Satellite Services Ltd.*)

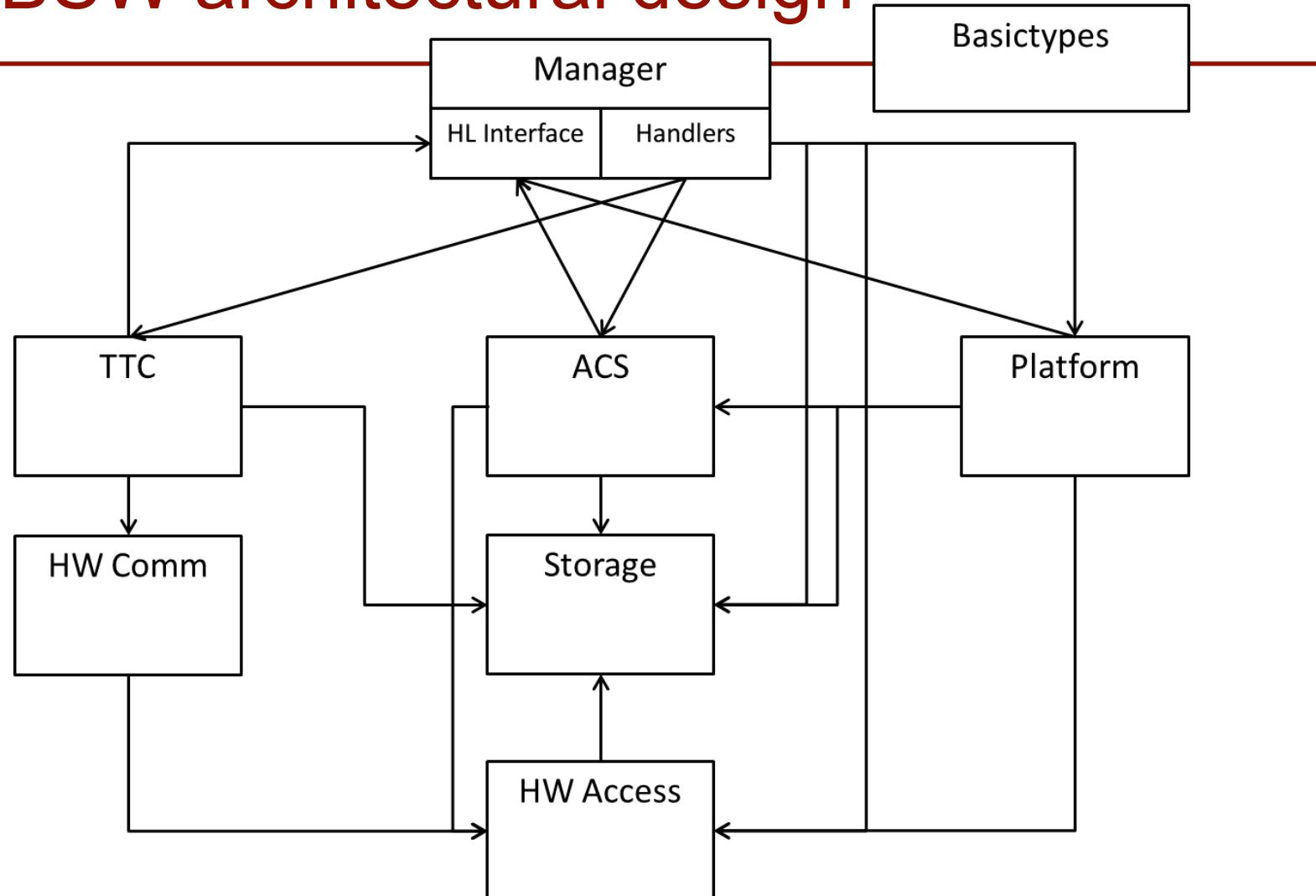
GS context



OBC operating modes

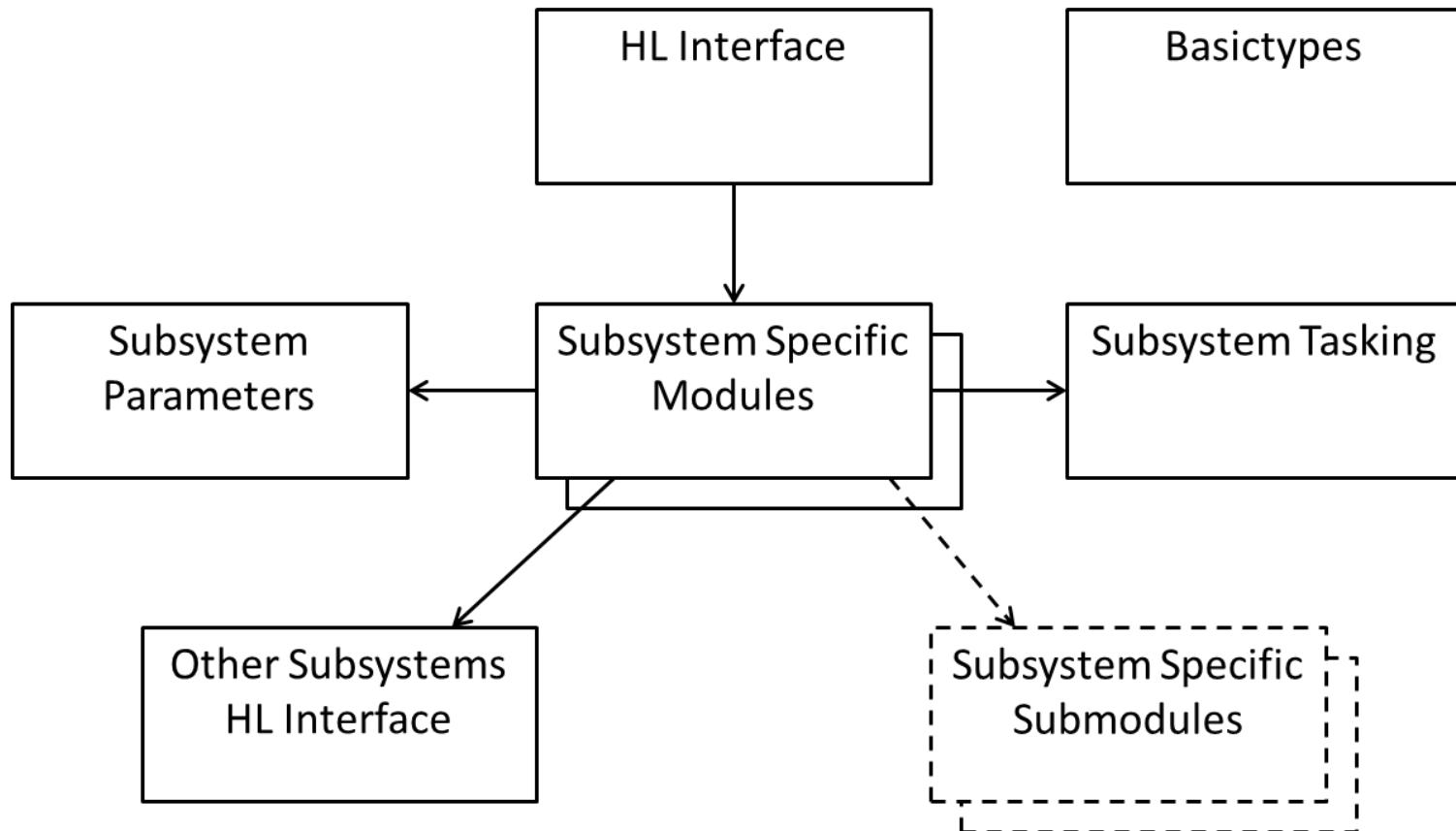


OBSW architectural design



© 2018 Jorge Garrido

Architecture of a subsystem



© 2018 Jorge Garrido

Sample code

- `with BasicTypes.TTC.TC;`
- `with Ada.Streams;`

- `-- @summary`
- `-- High-level interface of the manager subsystem.`
- `--`
- `-- @description`
- `-- This package provides the external interface of the Manager component.`
- `-- It manages the overall satellite operation. It manages the satellite state`
- `-- and handles events, errors and TCs`
- `--`
- **`package Manager.HLInterface is`**

- `-- Start the execution of the on-board software`
- **`procedure Start;`**

- `-- Notify an event to the manager`
- `-- @param the event to be handled`
- **`procedure Notify_Event (Event : in BasicTypes.Event);`**

- `-- Notify a TC to the manager`
- `-- @param CommandID Identifier of the command in the TC`
- `-- @param PacketDataField Contents of the TC`
- `-- @param PacketDataFieldLength Length of the contents of the TC`
- **`procedure Process_TC`**
- `(CommandID : in BasicTypes.TTC.TC.TC_Command_ID;`
- `PacketDataField : in Ada.Streams.Stream_Element_Array;`
- `PacketDataFieldLength : in Ada.Streams.Stream_Element_Offset);`

- `-- Get the current operating mode of the OBSW`
- `-- @return the operating mode`
- **`function Current_Mode return BasicTypes.Operating_Mode;`**

- **`end Manager.HLInterface;`**

Development tools

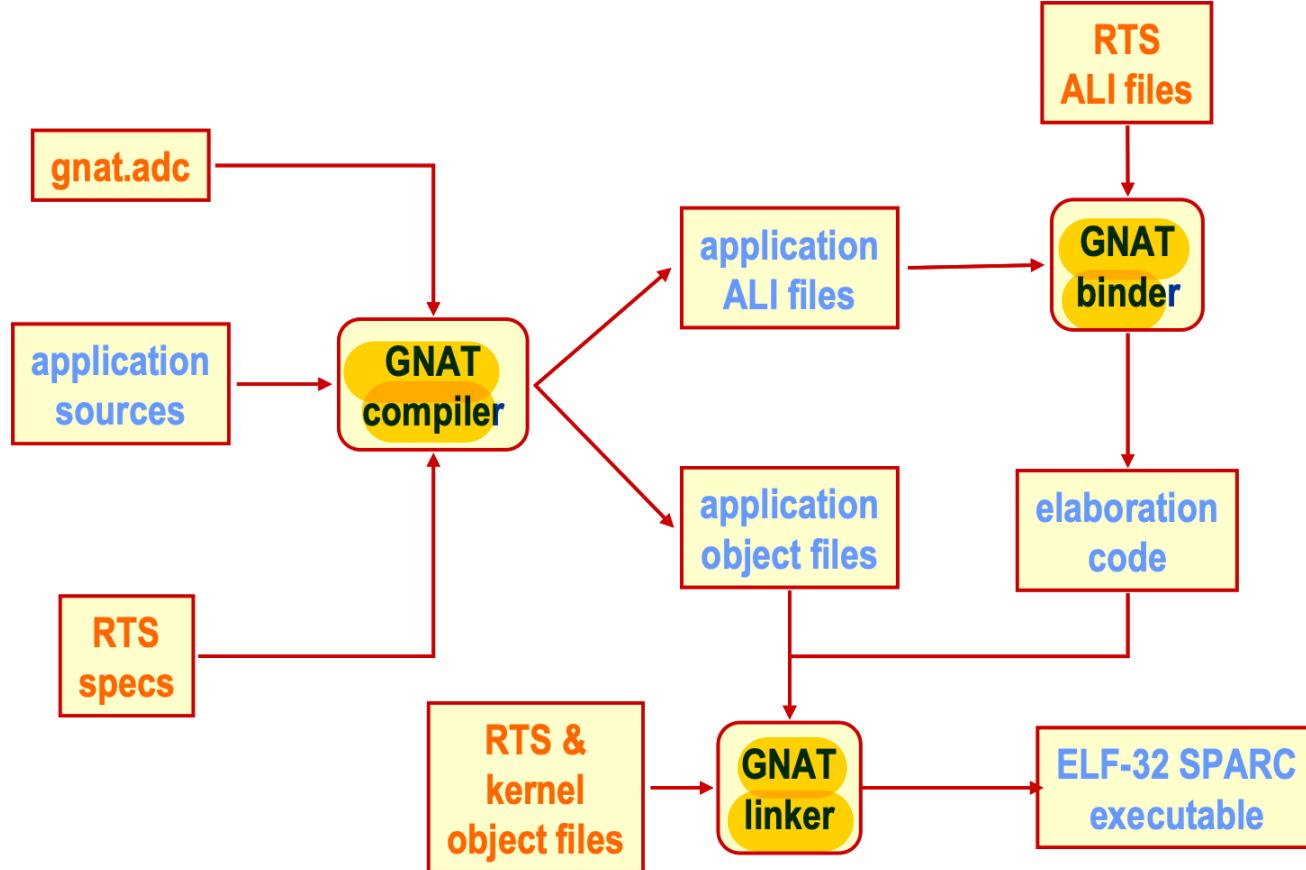
- Ada 2005 language
 - a few modules in C
 - automatically generated from Simulink
- Graphical programming environment
 - GPS: GNAT Programming System
- GNAT compilation chain
 - GNU Ada New York University Translator
 - Native compilers for development and high-level testing
 - Cross-compiler (GNU/Linux - LEON3) for production

GNAT Programming System

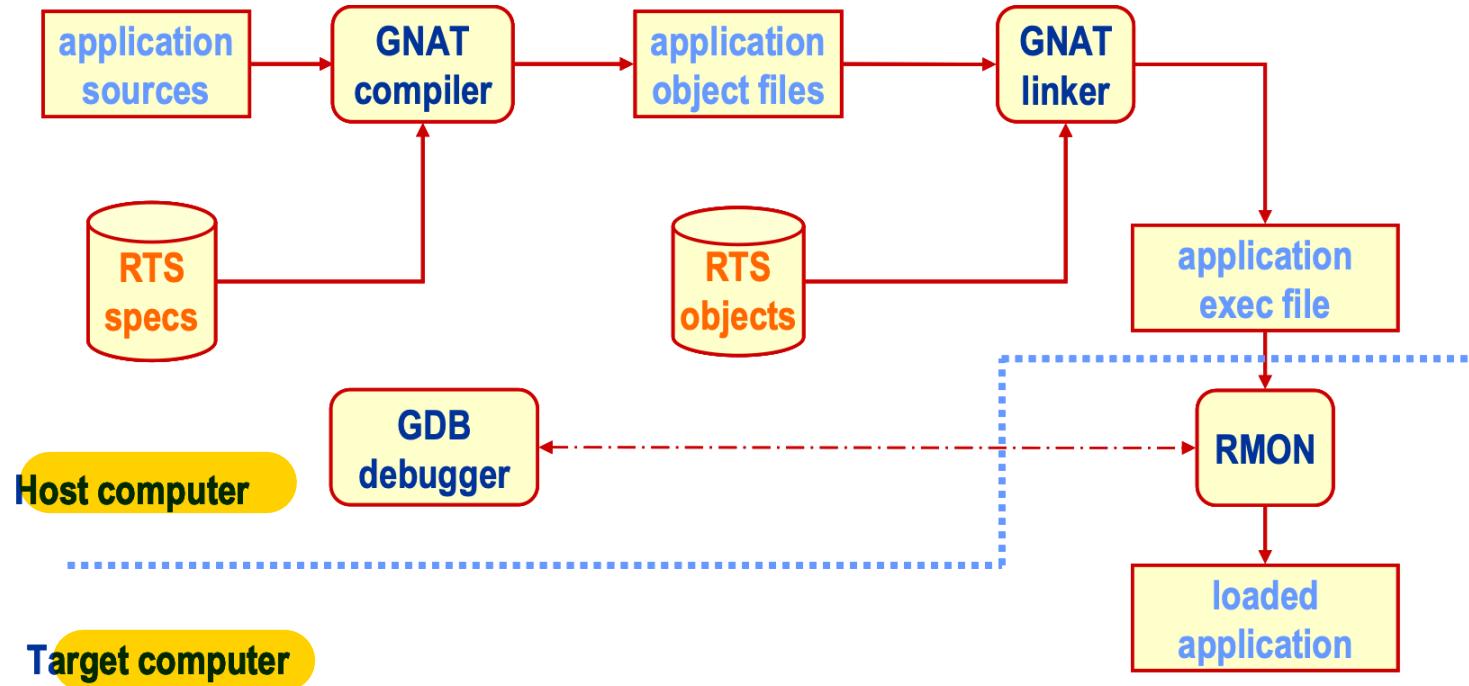
The screenshot shows the GNAT Programming System (GPS) interface with the following details:

- Toolbar:** File, Edit, Navigate, VCS, Project, Build, Debug, Tools, Window, Help.
- Project Browser:** Shows the project structure under "Project":
 - ADCS Test
 - Common
 - Experiment MTS Test
 - Experiment RW Test
 - Global
 - HWAccess Test
 - HWComm Sockets
 - Manager Test
 - src
 - manager-eventhandler.adb
 - manager-eventhandler.ads
 - manager-experimentsmanager.adb
 - manager-experimentsmanager.ads
 - manager-hlinterface.adb
 - manager-hlinterface.ads
 - manager-initialization.adb
 - manager-initialization.ads
 - manager-modemanager-operations
 - manager-modemanager-operations
 - manager-modemanager.adb
 - manager-modemanager.ads
 - manager-parameters.ads
 - manager-syncevent.adb
 - manager-syncevent.ads
 - manager-synctc.adb
 - manager-synctc.ads
 - manager-tasking.ads
 - manager-tc_operations.adb
 - manager-tc_operations.ads
 - manager-tchandler.adb
 - manager-tchandler.ads
 - manager.ads
 - obj
 - OBSW Test (root project)
 - Platform Test
 - Storage Test
 - TTC Test
- Code Editor:** Displays the source code for `manager-hlinterface.adb`. The code is Ada code defining a package body for `Manager.HLInterface`. It includes procedures for `Start`, `Notify_Event`, `Process_TC`, and `Current_Mode`.
- Status Bar:** Shows the line number 54:9 and other status indicators.

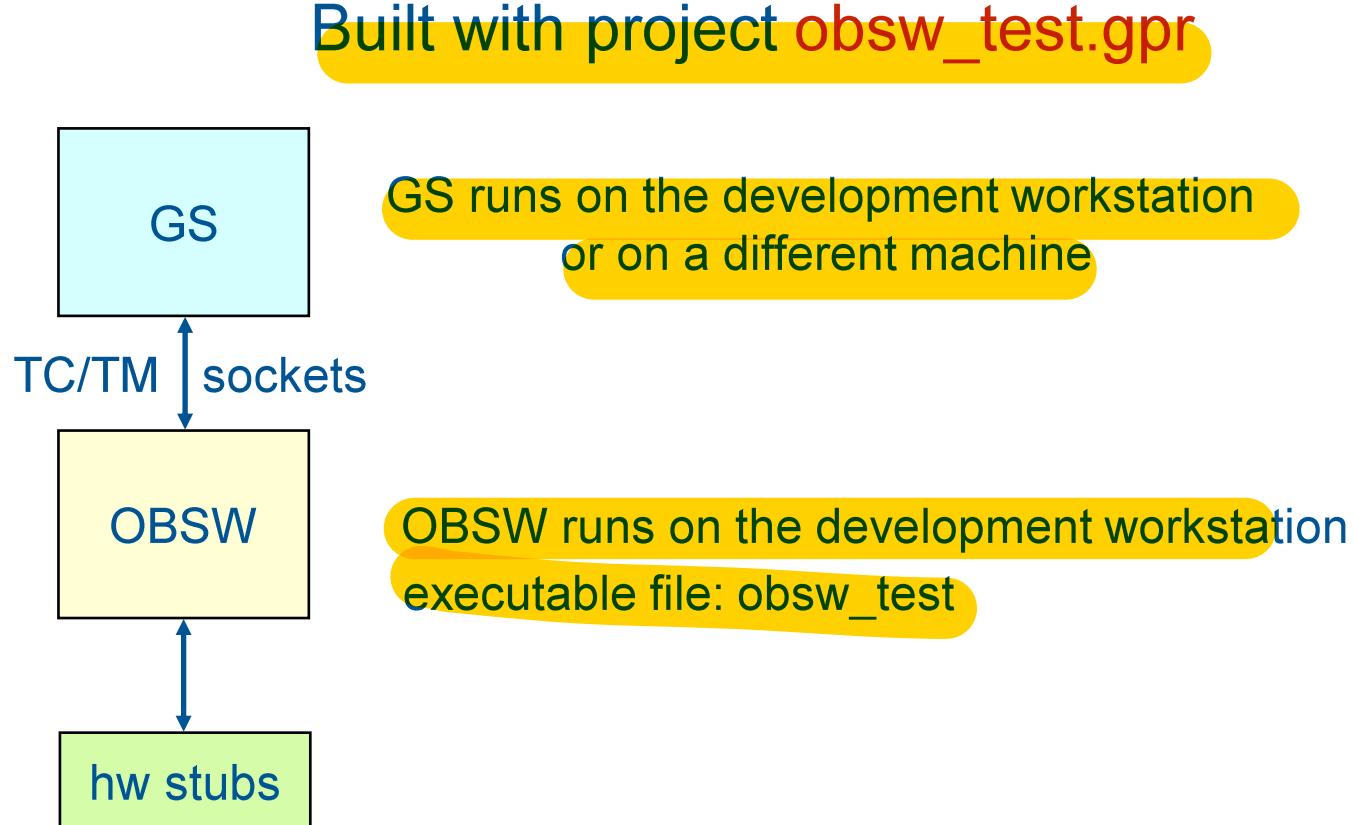
Cross-compilation process



Debugging tools

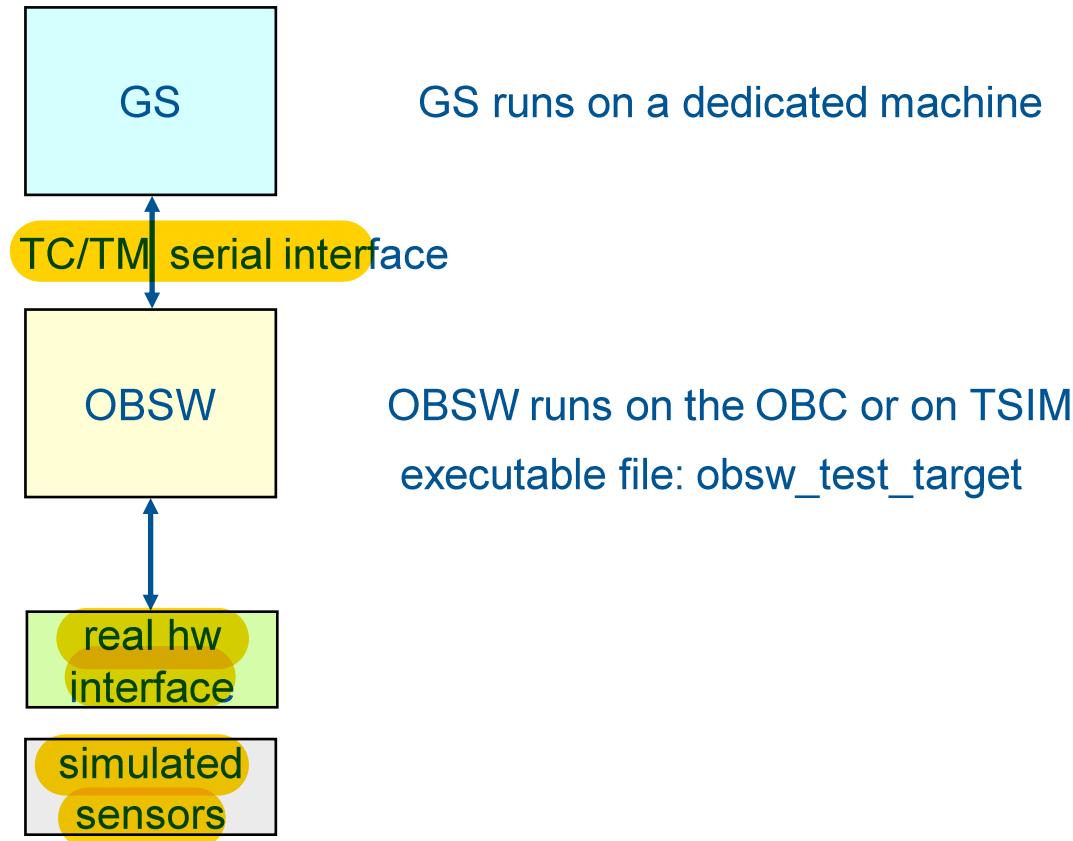


Testing (1)

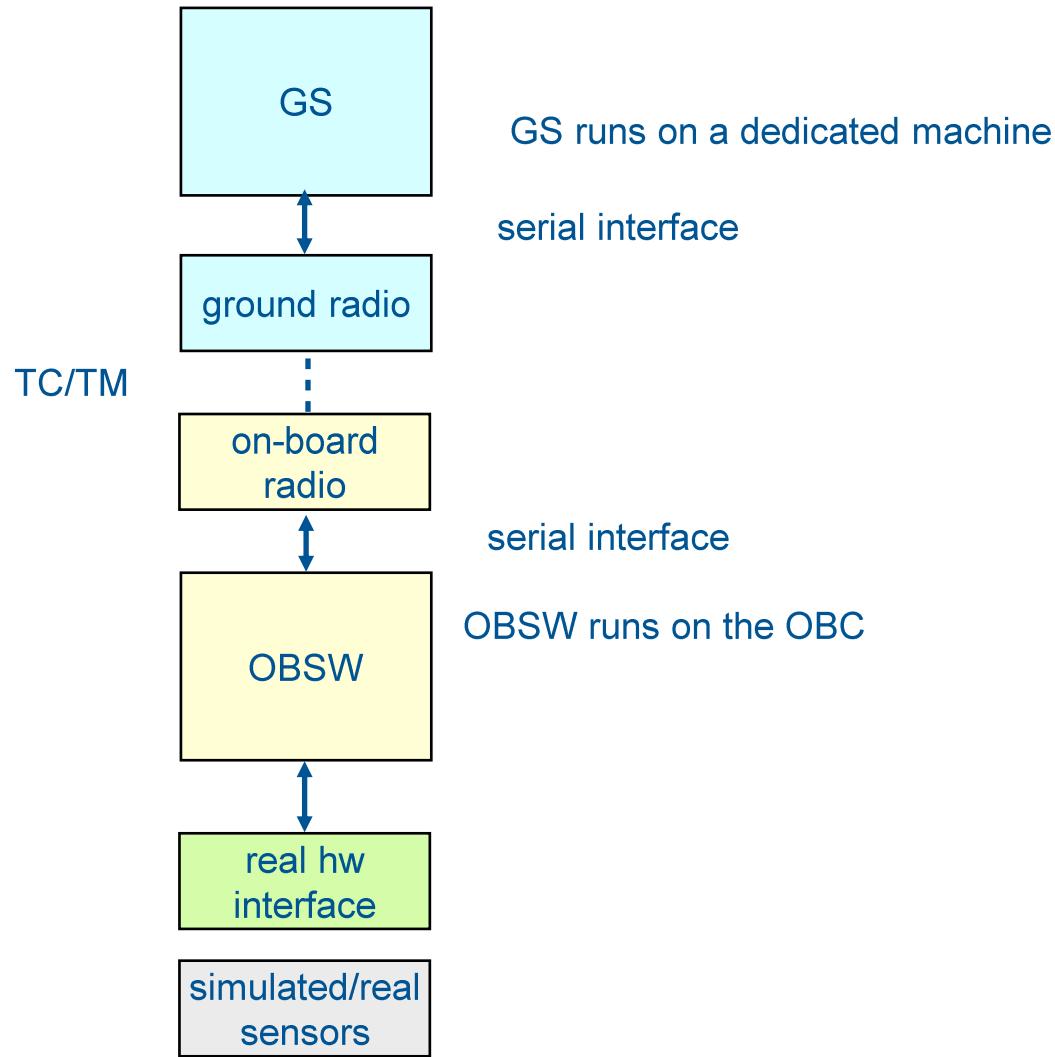


Testing (2)

Built with project `obsw_test_target.gpr`



Testing (3)

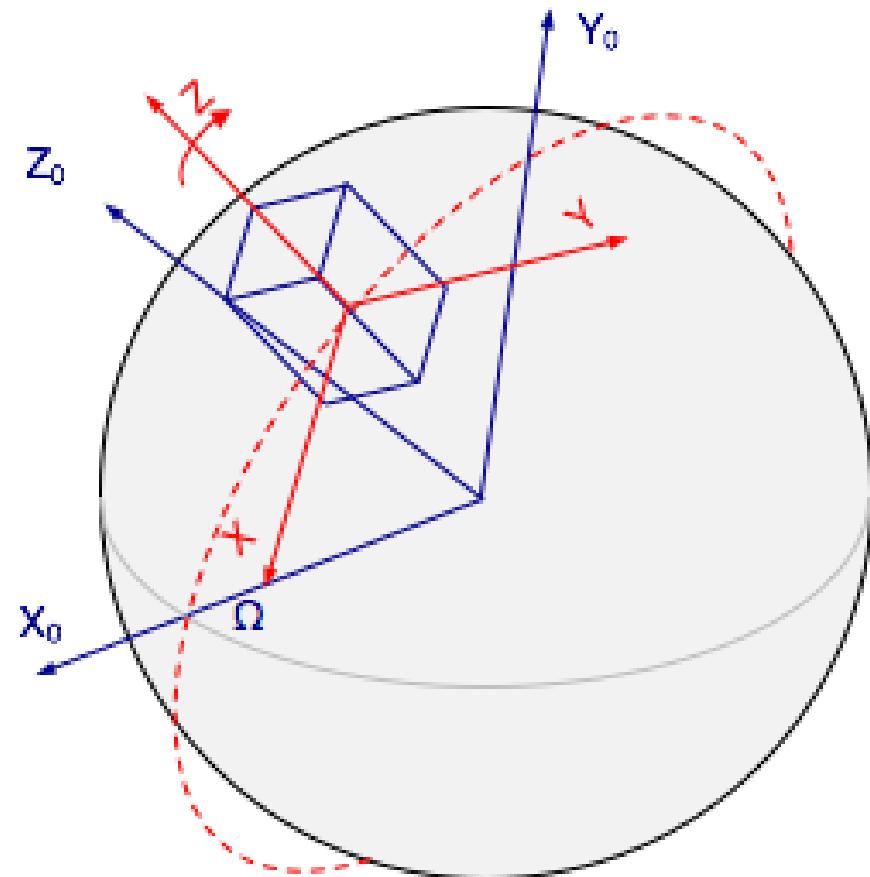


Attitude Control System Validation & Verification

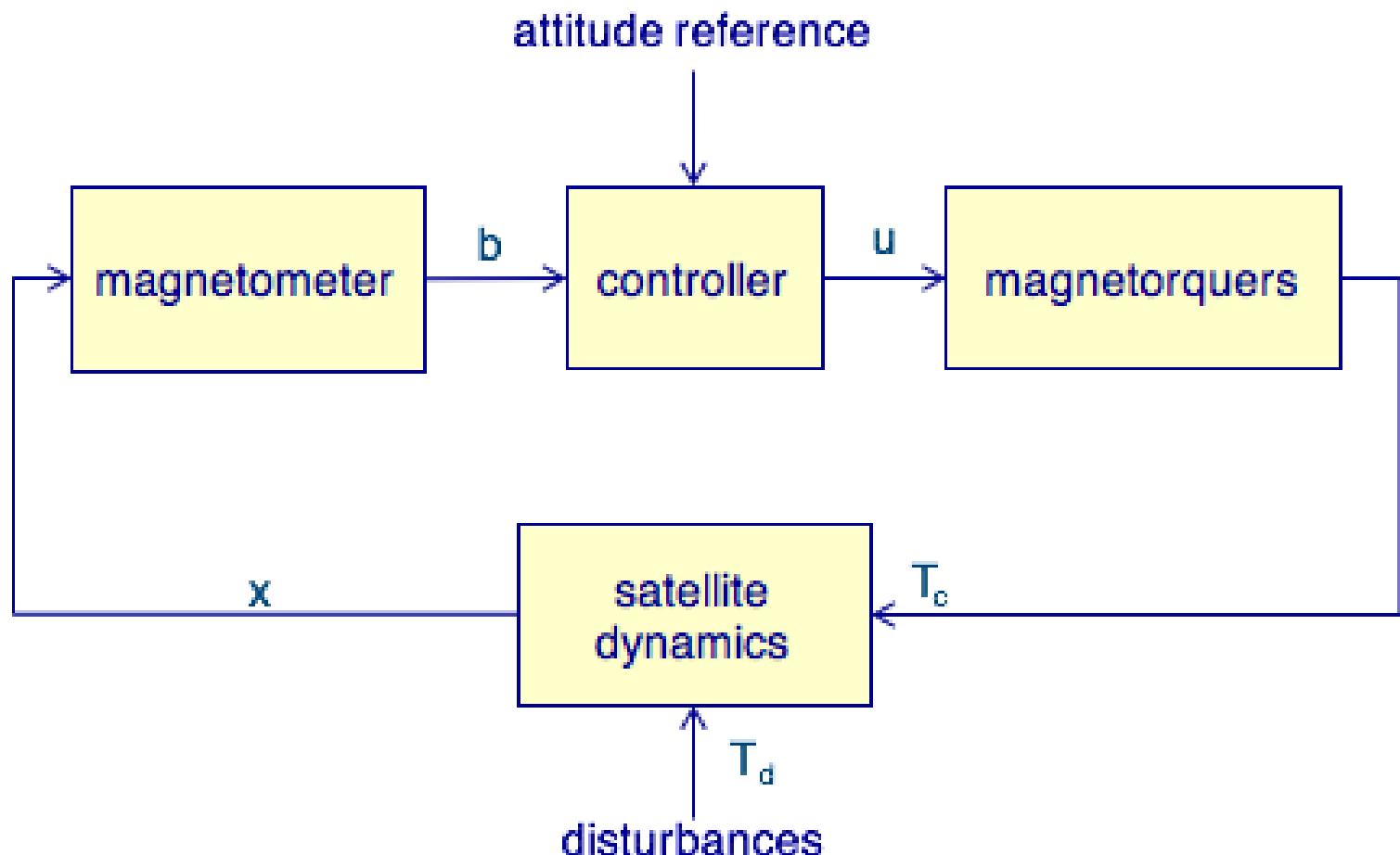
Attitude control frames

- Body reference frame XYZ
 - antenna on Z axis
- Orbital reference frame $X_0Y_0Z_0$
 - inertial for short intervals
- Attitude reference
 - align Z with Z_0
 - slow rotation around Z

$$\omega_d = [0 \ 0 \ \omega_r]^T$$



Magnetic attitude control



Attitude dynamics

- Euler's equation
 - \mathbf{I} : Inertia matrix
 - $\boldsymbol{\omega}$: angular velocity
- Control torque \mathbf{T}_c
 - generated by magnetorquers
- Disturbance torque \mathbf{T}_d
 - aerodynamic, magnetic, solar perturbations

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} = \mathbf{T}$$

$$\mathbf{T} = \mathbf{T}_c + \mathbf{T}_d$$

Magnetometers / magnetorquers

- Magnetometers provide a measurement of the Earth's magnetic field components in the body frame
 - \mathbf{b} : magnetic field vector
- Magnetorquers generate a control torque normal to the orbit

$$\mathbf{T}_c = \mathbf{m} \times \mathbf{b}$$

- \mathbf{m} : magnetic moment vector
 - proportional to the current in each magnetorquer coil
 - $\mathbf{m}_i = A\mathbf{N} \cdot \mathbf{I}_i$

Control law

- Classical b-dot control law

$$\mathbf{m} = -k\dot{\mathbf{b}}$$

- not valid for steady state control

- Enhanced control law used in UPMSat-2

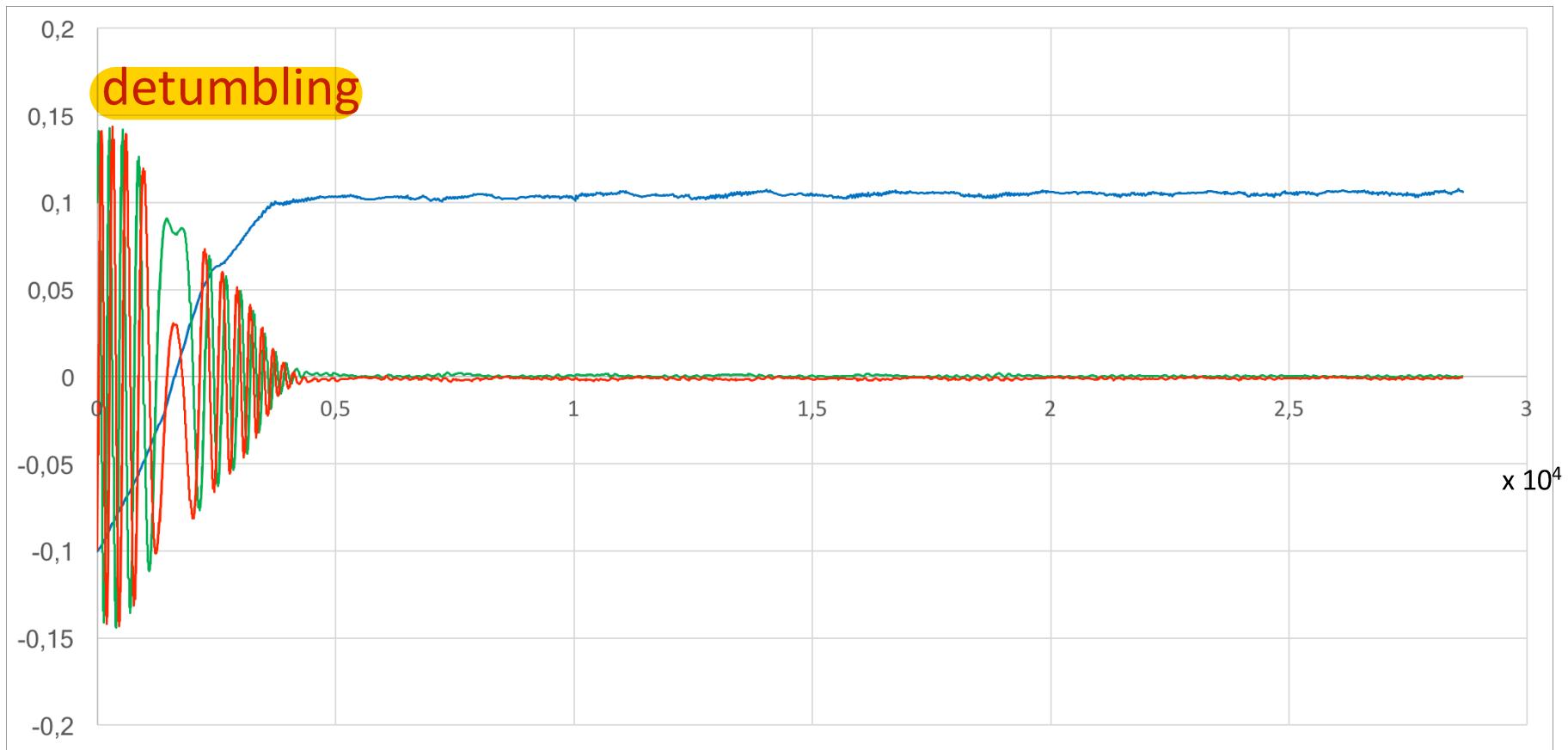
$$\mathbf{m} = -k(\dot{\mathbf{b}} + \boldsymbol{\omega}_d \times \mathbf{b})$$

- proved to be stable and robust
- consistent with UPMSat-2 desired attitude vector

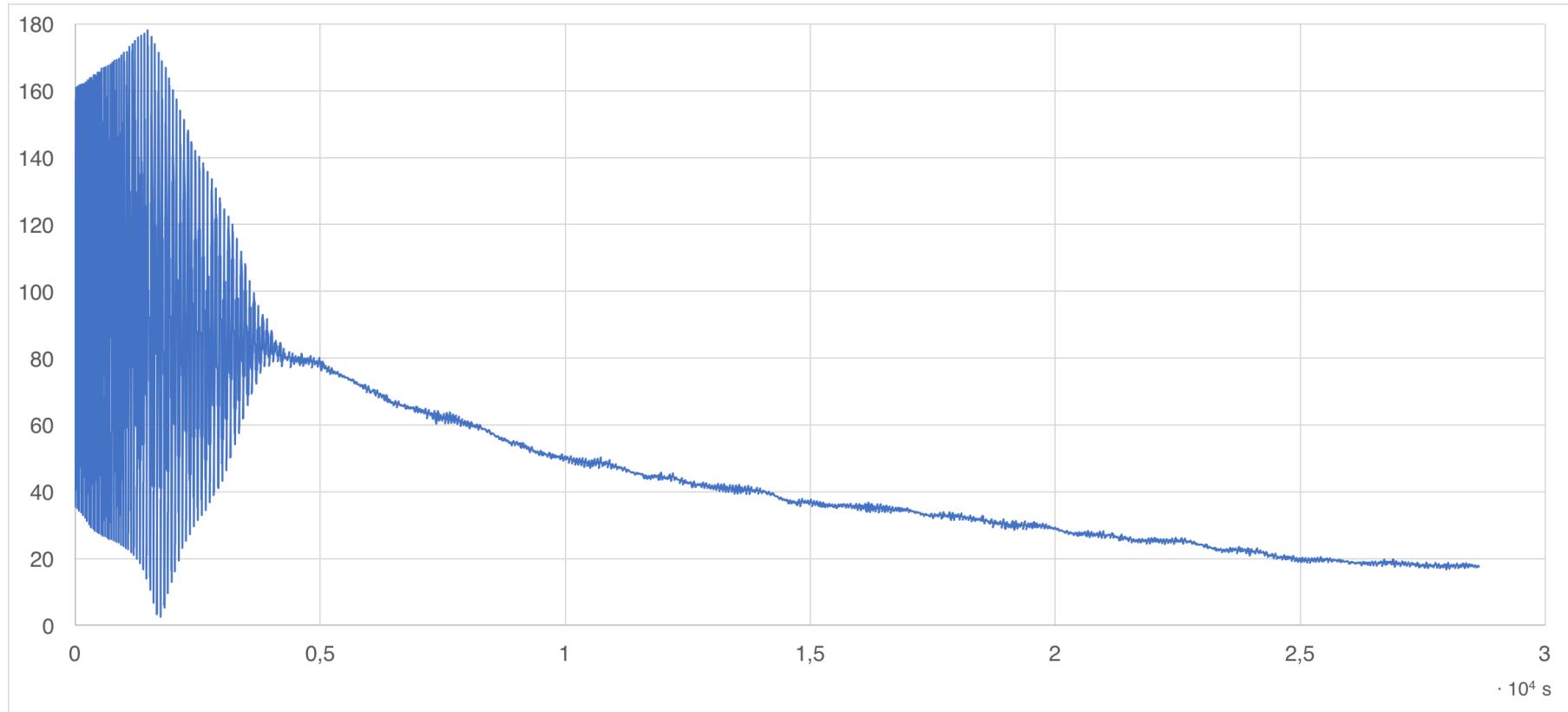
Cubas, J., Farrahi, A., and Pindado, S. (2015). Magnetic attitude control for satellites in polar or sun- synchronous orbits.

Journal of Guidance, Control, and Dynamics, 38(10), 1947–1958.

Angular velocity stabilization



Z-Z₀ alignment



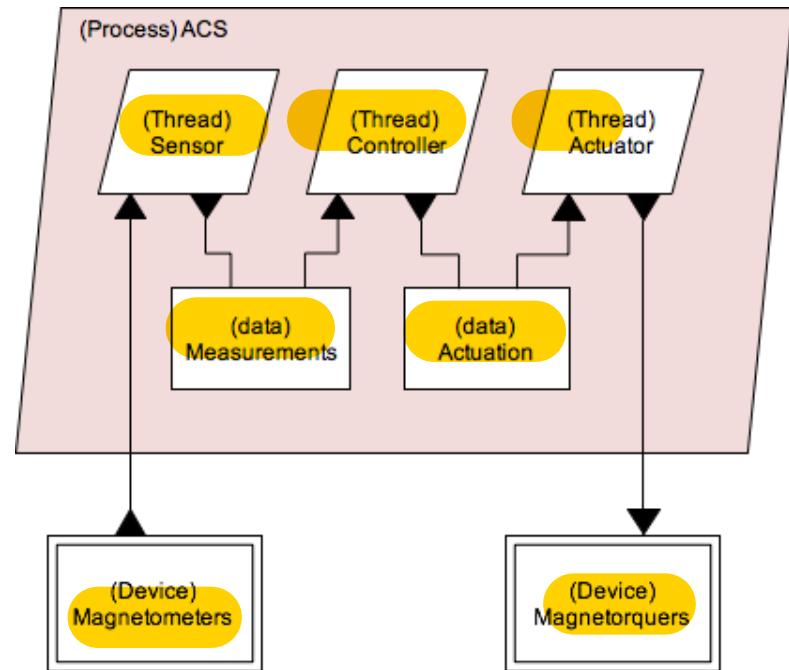
HW implementation

- Magnetometers
 - 2 redundant SSBV magnetometers + 1 experimental Bartington MGM
- Magnetorquers
 - 3 ZARM magnetorquers
- On-Board Computer (OBC)
 - LEON3 @ 20 MHz
 - 4 MB SRAM, 2 MB EEPROM

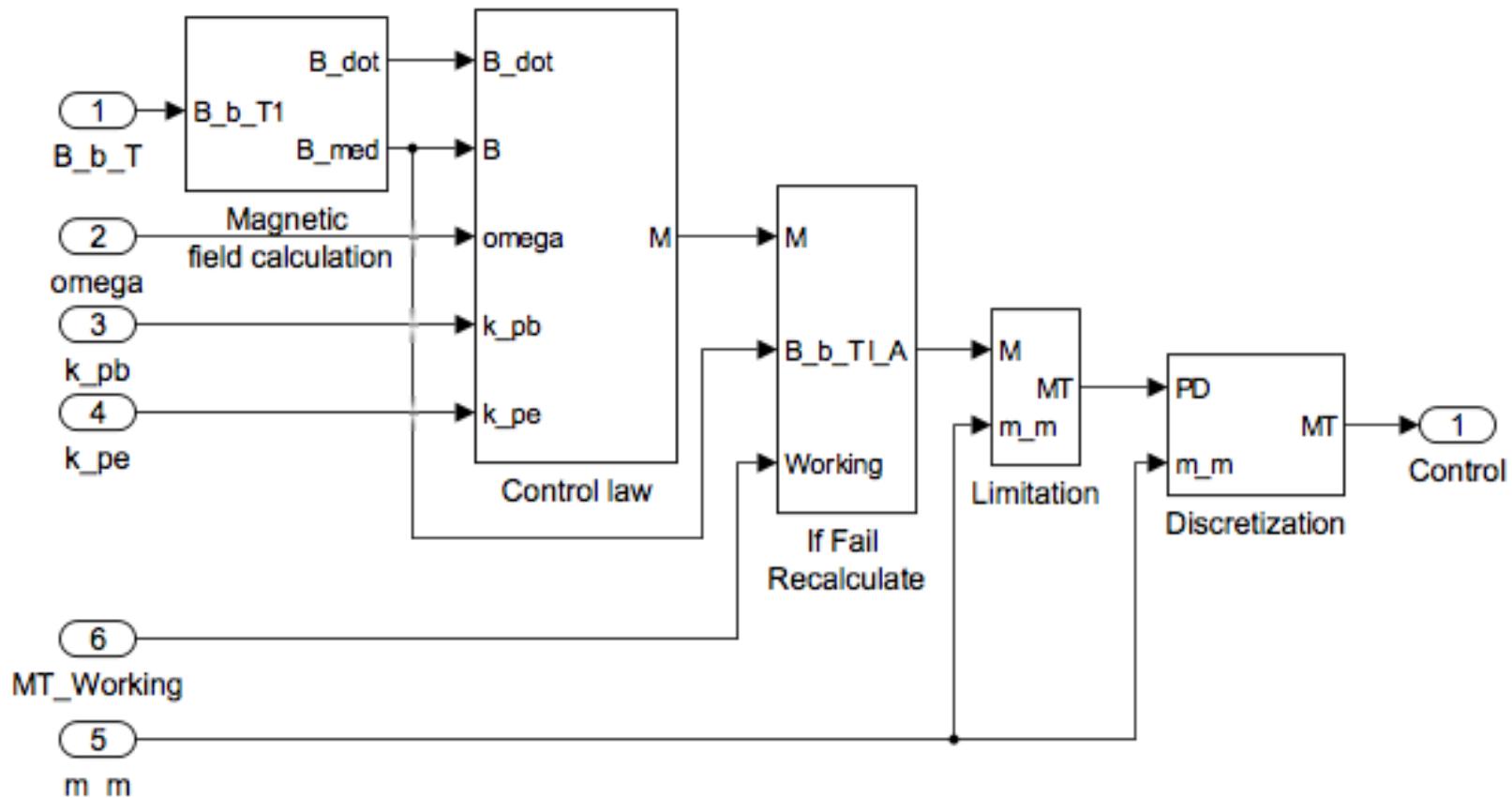


SW implementation

- Model-driven design
 - AADL
 - UML
 - Simulink
- Implemented in Ada
 - GNAT for LEON
 - ORK+ real-time kernel
- Controller code
 - generated from simulink model



Controller model



Technology Readiness Level (TRL)

The technical maturity of instruments and spacecraft sub-systems with respect to a specific space application are classified according to a "Technology Readiness Level" (TRL) on a scale of 1 to 9. ESA uses the ISO standard 16290 Space systems – Definition of the Technology Readiness Levels (TRLs) and their criteria assessment.

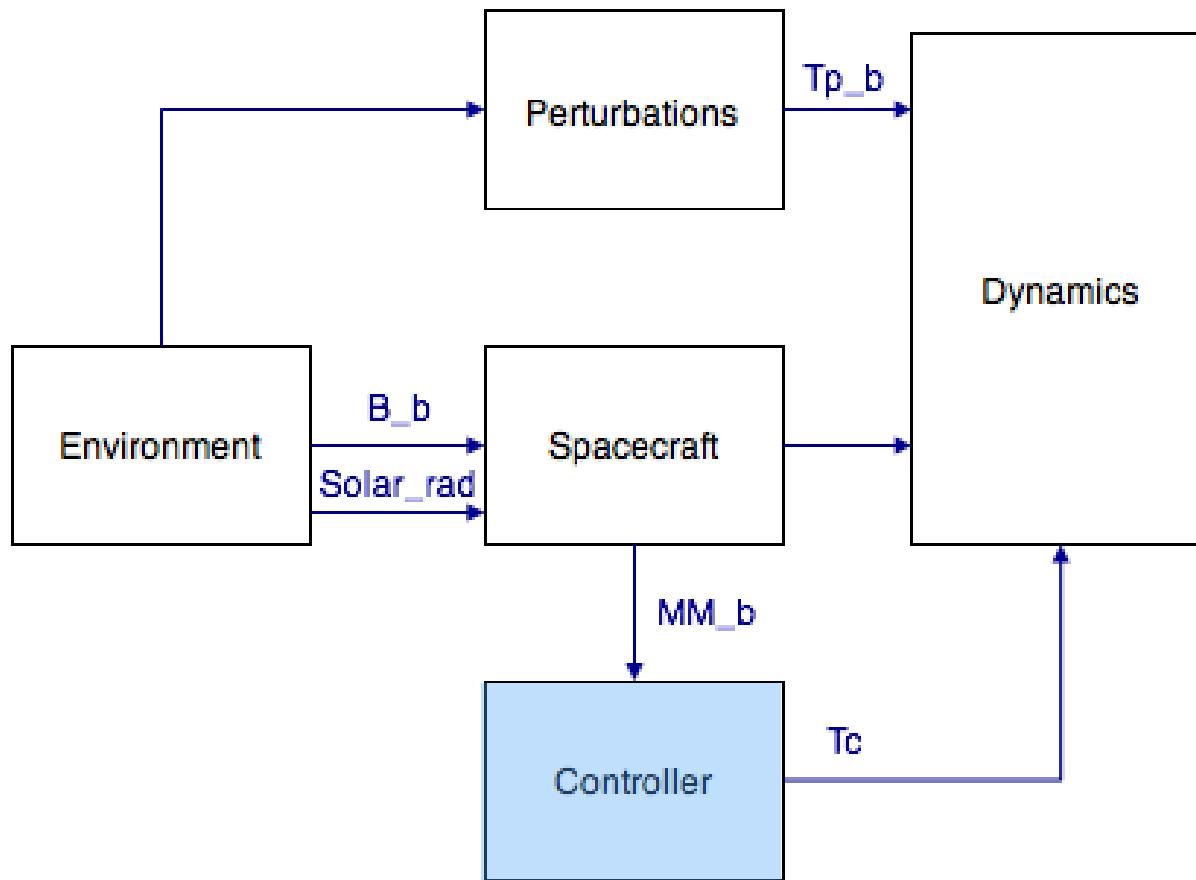
ISO Technology Readiness Level Summary

TRL Level Description

- 1 Basic principles observed and reported
- 2 Technology concept and/or application formulated
- 3 Analytical and experimental critical function and/or characteristic proof-of-concept
- 4 Component and/or breadboard functional verification in laboratory environment
- 5 Component and/or breadboard critical function verification in relevant environment
- 6 Model demonstrating the critical functions of the element in a relevant environment
- 7 Model demonstrating the element performance for the operational environment
- 8 Actual system completed and accepted for flight ("flight qualified")
- 9 Actual system "flight proven" through successful mission operations

<http://sci.esa.int/sci-ft/50124-technology-readiness-level>

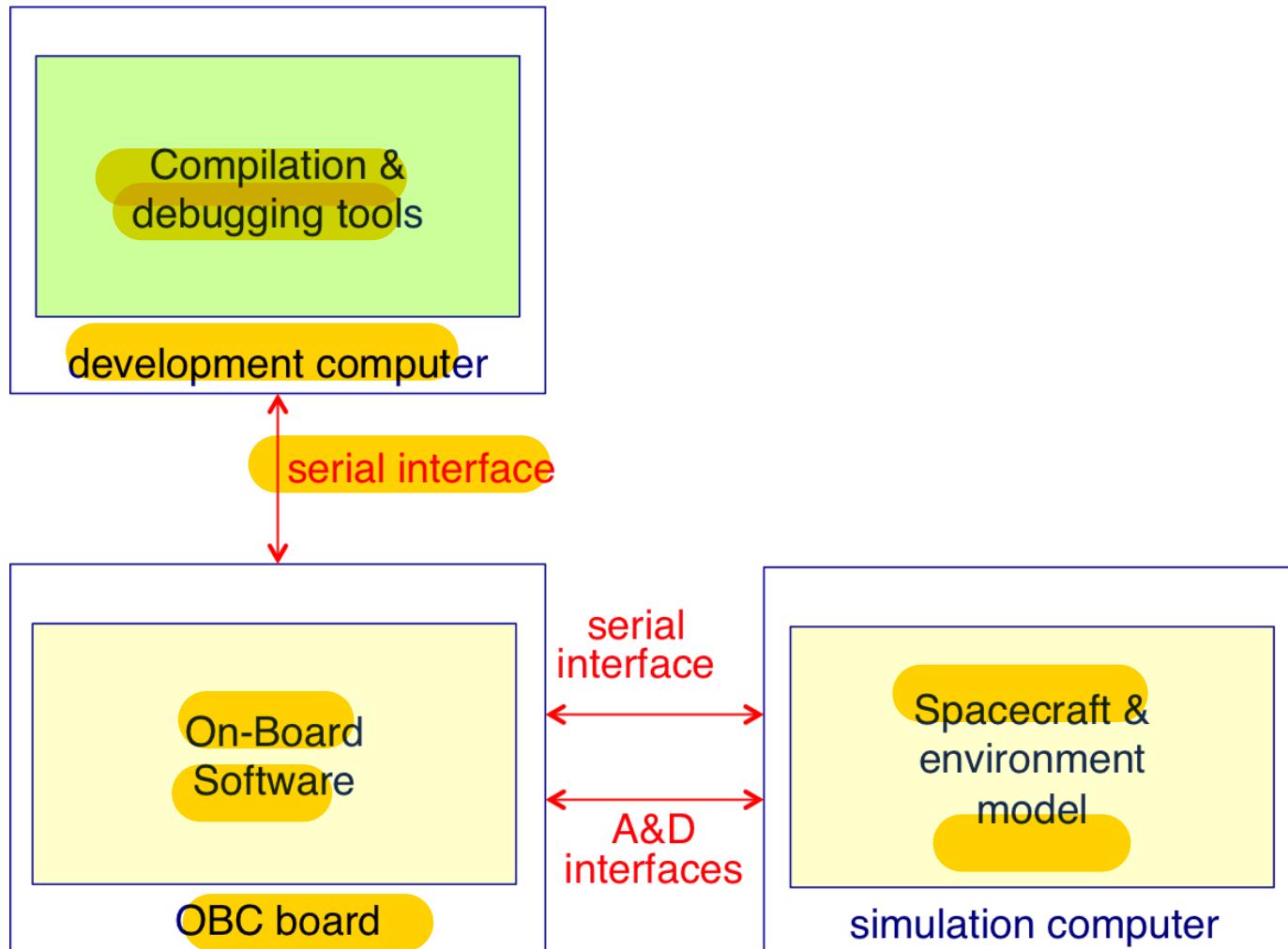
ACS validation



Validation steps: Model-in-the-loop

- Preliminary validation of control algorithm: this first validation phase uses a model of the ACS, together with models of the space environment and the spacecraft dynamics, to assess the validity of the control law and the design parameters.
- It is usually carried out by a control engineer using a simulation tool.
- Simulink is used for UPMSat-2 ACS.

Software Validation Facility



Validation steps: Processor-in-the-loop

- A further step to full validation is running the control software on actual hardware, while still using simulation models of the sensors, actuators, and spacecraft dynamics.
- It is needed a software validation facility (SVF), which uses an auxiliary computer, linked to the OBC by a serial line, to run a simulation model of the Earth's magnetic field and the satellite dynamics. In this way, engineering values (Nm and T) can be interchanged.
- The code of the controller algorithm was generated by using Simulink Code Generator tool and compiled and linked with the on-board software with GNAT for LEON development environment.
- Functional & real-time behaviour can be validated.

Validation steps: Hardware-in-the-loop

- The next step is to include a model of the actual magnetometers and magnetorquers in the simulation model, and connect their inputs and outputs to the OBC hardware by means of its analog and digital input and output lines.
- This simple kind of HIL setup allows the operation of input/output devices, as well as signal conditioning hardware and software, to be validated.
- Full HIL validation, though, requires using the real magnetometers and magnetorquers hardware and, ultimately, the whole satellite structure, on a setup which enables the attitude of the spacecraft to evolve in a free fashion.

Arquitectura de hardware

Arquitectura de hardware

- En ingeniería de sistemas se usa el término de forma análoga al de arquitectura de software.
 - Se refiere a las estructuras de alto nivel, su funcionamiento y sus relaciones e interacciones.
 - Es decir su diseño y descomposición en módulos que agrupan funcionalidades similares dentro del sistema.
 - Los módulos deben tener interfaces consistentes y bien definidas para poder interconectarse.

Módulos de hardware

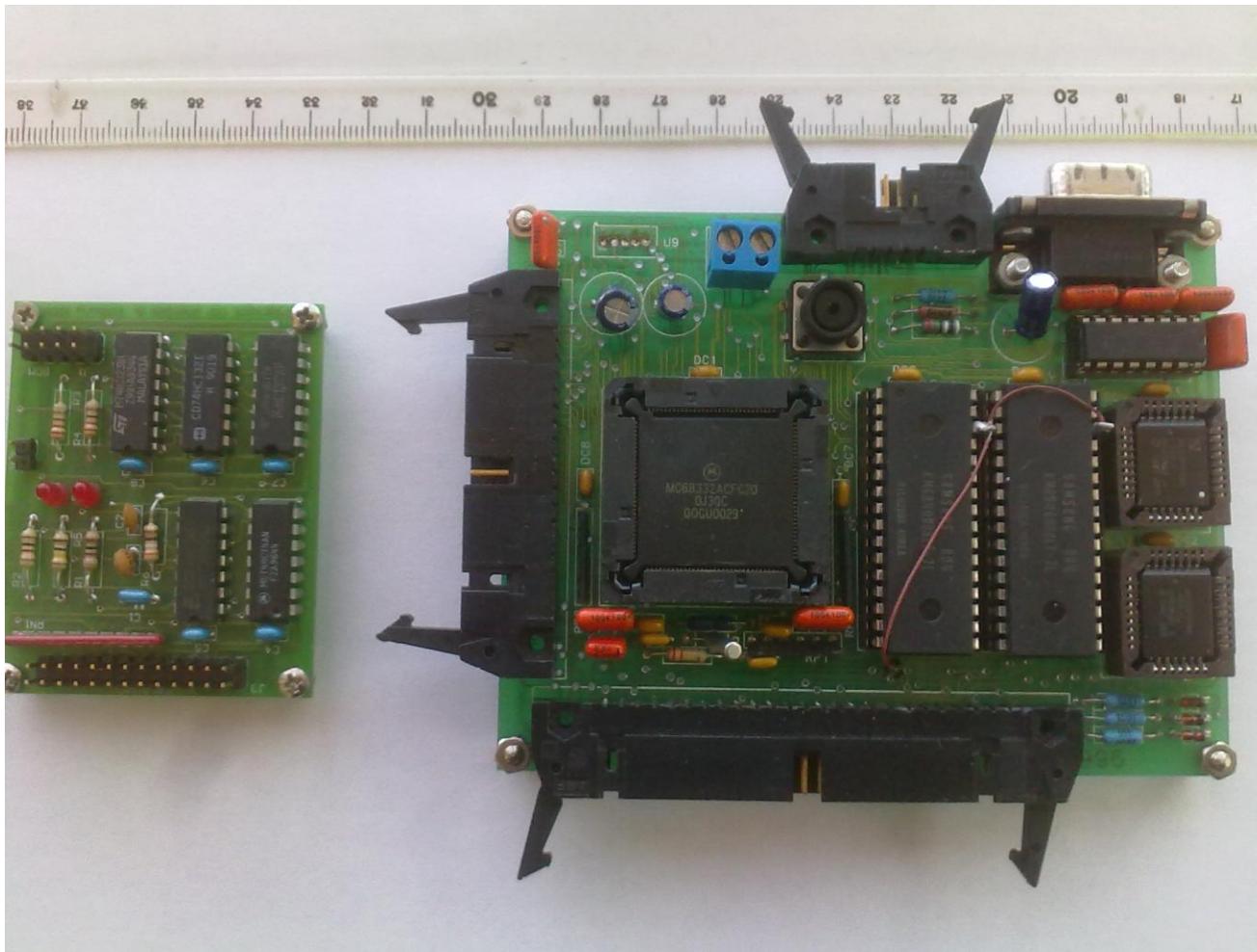
- En el caso del hardware, se elige entre los módulos disponibles en el mercado en forma de **Integrated Circuits (IC)**:
 - Procesadores, memorias, módulos de entrada/salida, multiplexores, codificadores, etc.
- También es posible construir en base a módulos más complejos:
 - Tarjetas para computadores modulares basadas en diferentes normas y estándares:
 - cPCI, VMEbus, PC104, etc.
- Y construir módulos específicos mediante lenguajes de descripción de hardware (**HDL**):
 - ASIC, FPGA, semi-custom, full-custom, etc.

Circuitos integrados

IC, chips o micro chips están compuestos de un trozo pequeño de material semiconductor con circuitos electrónicos metidos en una cápsula de plástico o cerámica y con conductores metálicos (*pines*) apropiados para su inserción en circuitos impresos.

- Hoy día son muy complejos con hasta miles de millones de transistores y centenares de *pines*.
- En base a ellos se pueden construir sistemas computacionales.
 - Altos costes de desarrollo y mantenimiento.
- Se fabrican en familias lógicas:
 - M68000, 7400, intel 8088/82xx, etc.

Diseño con circuitos integrados



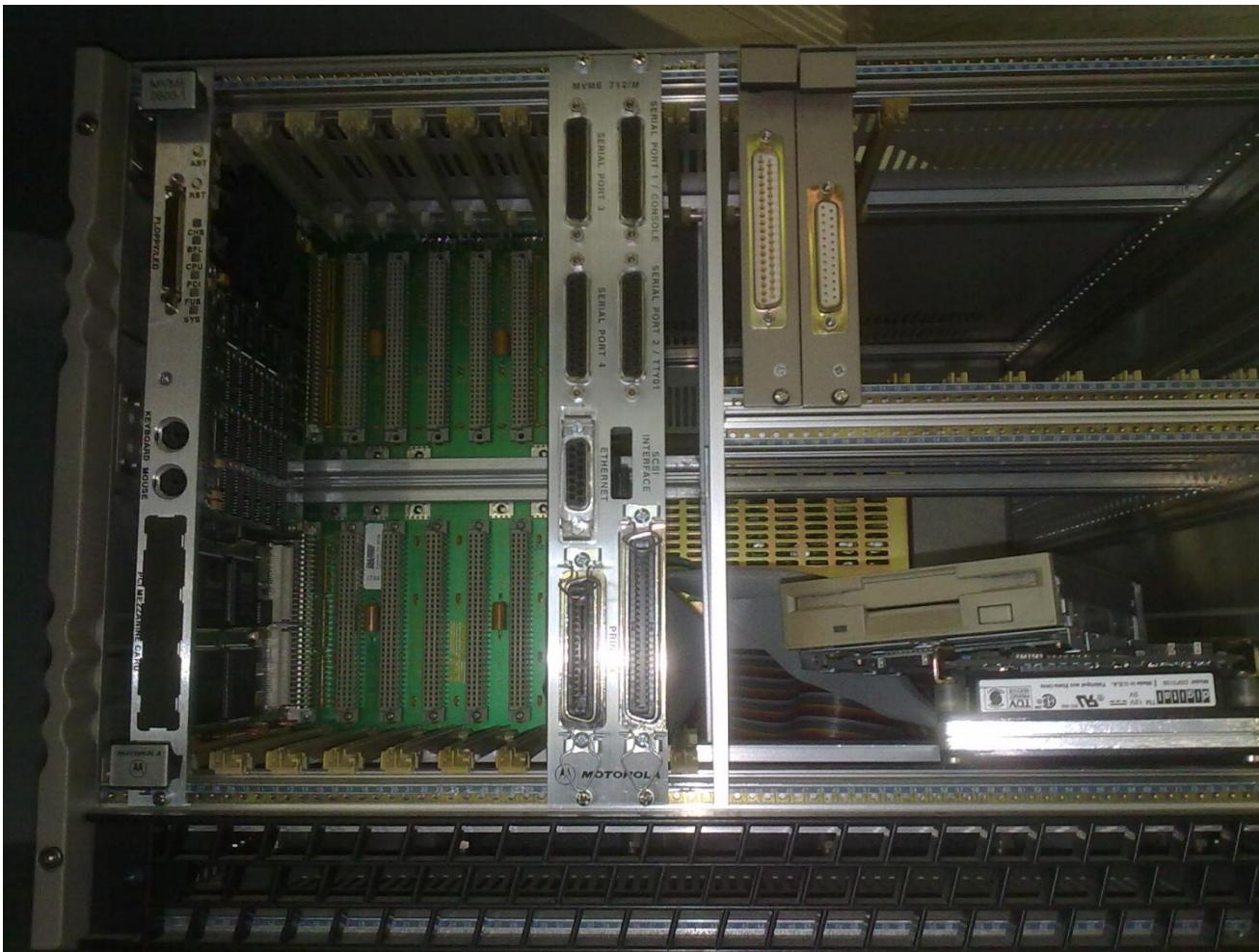
Computadores modulares

- Computadores construidos en base a placas con distintas funcionalidades: CPU y memoria, unidades de entrada/salida.
 - Orientado a su uso industrial.
- Existen multitud de estándares industriales que definen:
 - Normas mecánicas
 - Eléctricas
 - Lógicas
- Multitud de fabricantes proporcionan catálogos de diversidad de placas y módulos para las normas más populares:
 - cPCI, PC104, VME, ...

Computadores modulares

- Se pueden construir “a medida” combinando varias placas para cubrir las necesidades del sistema.
 - Costes de desarrollo reducidos:
 - Selección de componentes y configuración del sistema.
 - Proporciona mantenibilidad ya que se puede añadir funcionalidad con nuevos módulos.
 - Puede resultar difícil encontrar repuestos a medio plazo.
 - Es difícil que el sistema se ajuste exactamente a los requisitos. El sistema resultante suele tener mayor funcionalidad, volumen, consumo eléctrico,...

Computadores modulares



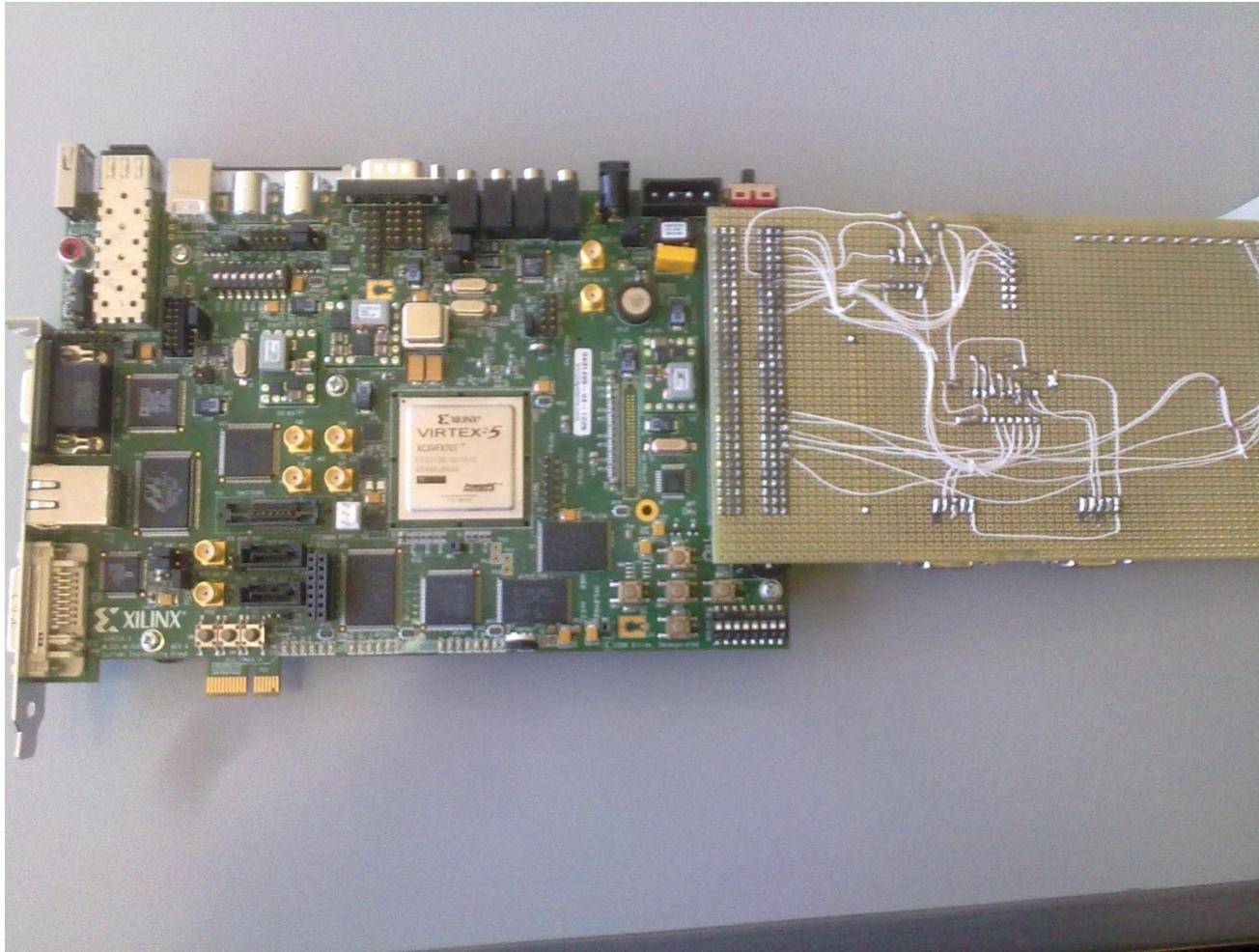
Hardware configurable

- Está basado en Programmable Logic Devices (PLD)
 - Son ICs con lógica configurable y biestables con interconexiones programables. De este modo, se pueden configurar para proporcionar distintas funciones.
- Las Field-Programmable Gate Array (FPGA) es un tipo de PLD con bloques lógicos capaces de realizar funciones complejas.
 - Procesadores, memorias, módulos de entrada/salida y funciones “propias” del software.

Hardware configurable

- El coste de desarrollo es reducido si se utilizan bibliotecas de “IP Cores”:
 - Bibliotecas con componentes codificados en un HDL.
 - Configurar y sintetizar la unidad.
- Es posible desarrollar módulos específicos para cubrir la funcionalidad necesaria (hardware o software).
- Permite mantenibilidad y reconfiguración.
- Es necesario diseñar y construir la placa con el PLD y sus interfaces con el resto del sistema.

Hardware configurable



Arquitectura de hardware

- Es común utilizar estos tres paradigmas en una implementación:
- EBOX del UPMSat-2
 - OBC basada en una FPGA
 - Construída en base a ICs
 - DAS, RADIO y PDU construídas en base a ICs y componentes específicos (modem, relés)
 - Tarjetas con norma mecánica simple-Europa, contenidas en un *rack* de 3 unidades e interconectadas mediante un *back-plane*.

Hardware requirements specification

- Captura los requisitos que atan al hardware del sistema o subsistema.
- Tiene una estructura similar al software requirements specification.
- No existen estándares específicos para requisitos de hardware, aunque sí para requisitos de sistema y requisitos de software:
 - 830-1998 - IEEE Recommended Practice for Software Requirements Specifications
 - 1233-1998 - IEEE Guide for Developing System Requirements Specifications
 - 29148-2011 - Systems and software engineering - Life cycle processes - Requirements engineering

Ejemplo de requisitos hardware UPMSat-2

4.2.2 Hardware environment

The software will run on a dedicated on-board computer (OBC) with the following characteristics:

- FPGA-based LEON3 CPU
- 4 MB SRAM
- 2 MB EEPROM
- Serial interfaces: RS-232, RS-422, SPI, I2C
- Device interfaces: 64 analog input channels, 112 digital I/O signals

Ejemplo de requisitos hardware UPMSat-2

- TS-37 Persistent storage

The system shall provide persistent storage for configuration parameters, system state, and data and event log.

The amount of persistent storage available for data and event logging will be enough for a 12 hour period of log data to be stored.

- TS-38 Separation timer

A hardware timer will be activated at the time when separation from the launcher takes place. When the timer expires, the OBC is powered on, and the software starts loading and executing in initialization mode.

- TS-41 Mission clock

A hardware clock shall keep track of the time elapsed since separation.

Ejemplo de requisitos hardware UPMSat-2

Estos requisitos específicos derivan de requisitos de alto nivel:

- Se pretende usar un procesador estándar de la ESA.
- Los requisitos económicos impiden usar un procesador con calificación de vuelo.
- La baja órbita y la duración de la misión permiten usar componentes con menor calificación.
- Los requisitos legales y económicos implican usar una biblioteca de IP Cores con licencia GPL.

Como resultado se decidió usar la biblioteca GRLIB-GPL que contiene el IP Core del procesador LEON3 y sintetizarlo en una FPGA de calificación militar.

Ejemplo de requisitos hardware UPMSat-2

- La cantidad necesaria de memoria RAM se calculó en base a la experiencia previa del grupo con estos procesadores.
- El número de señales digitales y entradas analógicas se calcula a partir de los sensores y actuadores que se embarcarán en el satélite.
- El número y tipo de buses de comunicación viene dado por los interfaces de otros dispositivos embarcados:
 - UARTs: módem, rueda de inercia, carga y depuración de software.
 - SPI para el ADC elegido.
 - I2C para el módulo de reloj de misión elegido.

Ejemplo de requisitos hardware UPMSat-2

La cantidad de memoria no volátil se elige de acuerdo al período y duración de la cobertura, la velocidad de transmisión de la radio y el protocolo de comunicaciones.

- 10 minutos cada 12 horas
- 19200 bits/segundo
- Se pueden transmitir: 1.440.000 bytes
- 33% de información adicional en el protocolo de comunicaciones.
- Se deben almacenar: 950.400 bytes
- Se necesitan 2 Mb porque además debe almacenar el código de la aplicación e información de configuración.

Bloques de la EBOX

- Compuesta por 4 placas interconectadas por un bus de panel posterior (*back-plane*).
 - PSU: Power Supply Unit
 - PDU: Power Distribution Unit
 - DAS: Digital to Analog Subsystem
 - OBC: On-Board Computer

Summary

- OBDH hardware and software must comply with hard requirements
 - environment: radiation, temperature, acceleration
 - real-time
 - fault-tolerance
- Appropriate architectures, components and V&V techniques must be chosen