KiboCUBE Academy

Lecture 19

Introduction to CubeSat System Integration and Electrical Testing

Meijo University

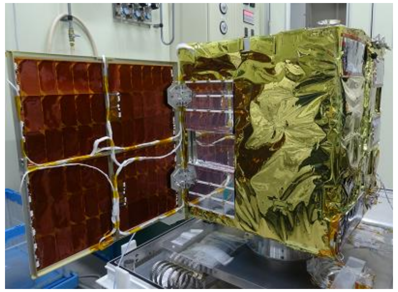
Department of Vehicle and Mechanical Engineering

Associate Professor Dr. Eng. Kikuko Miyata

This lecture is NOT specifically about KiboCUBE and covers GENERAL engineering topics of space development and utilization for CubeSats.
The specific information and requirements for applying to KiboCUBE can be found at:
https://www.unoosa.org/oosa/en/ourwork/psa/hsti/kibocube.html

Kikuko Miyata, Dr. Eng.

© Kyushu University



© Nagoya University

© The University of Tokyo

Major Positions:

2011 - Researcher, Next generation Space system Technology Research Association(NESTRA).

2014 - Postdoctoral fellow(-2016 Jul.). Designated assistant professor(2016 Aug.-Nov.), Assistant professor(2016 Dec.- 2020 Mar), Nagoya University.

2020 - Associate professor, Meijo University.

Research Topics:

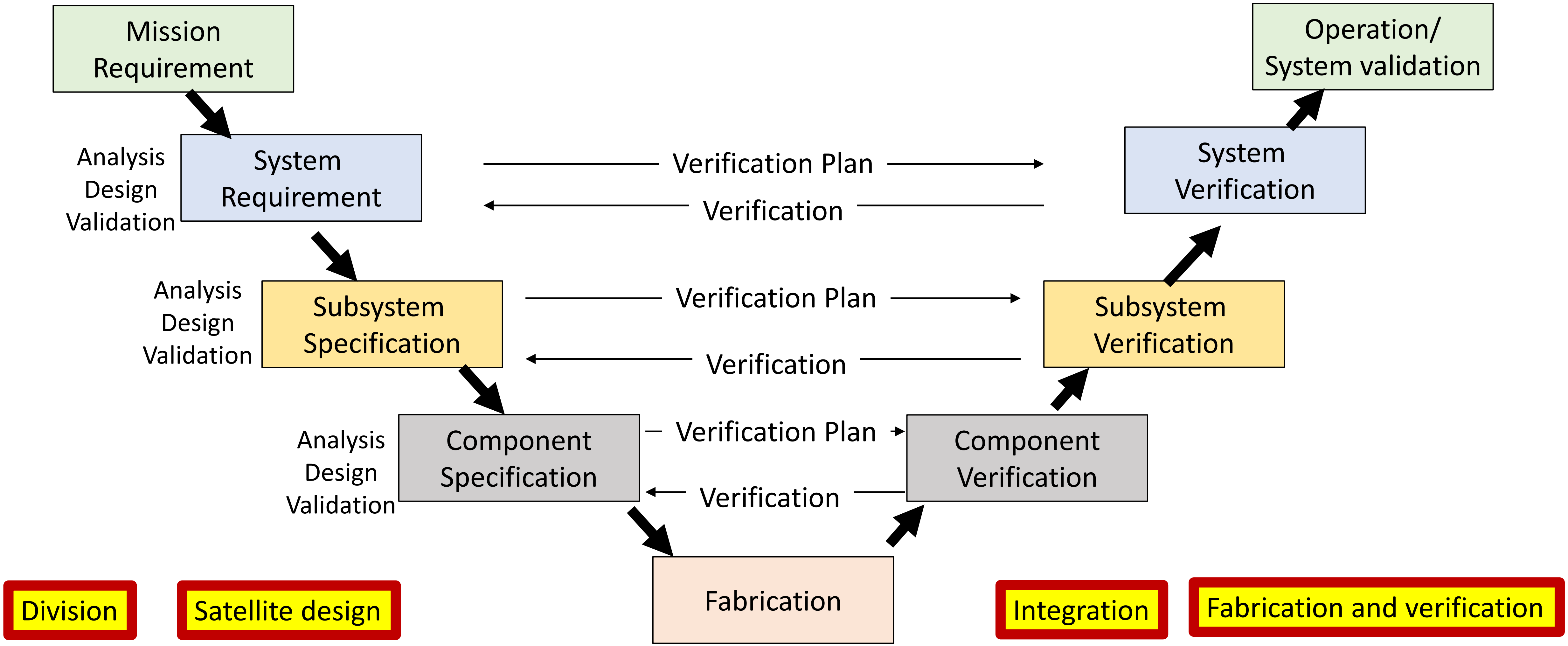
Small spacecraft system and related technology

Contents

1. Introduction to Lecturers

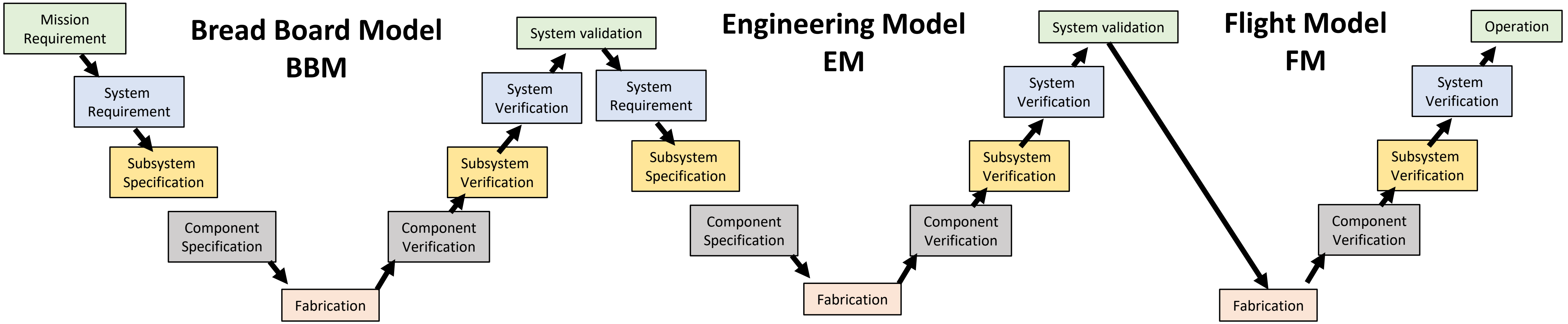
1.1 Overview of satellite development

- **Design and fabricate the system to achieve mission requirements**



1.1 Overview of satellite development

- **Development life cycle**



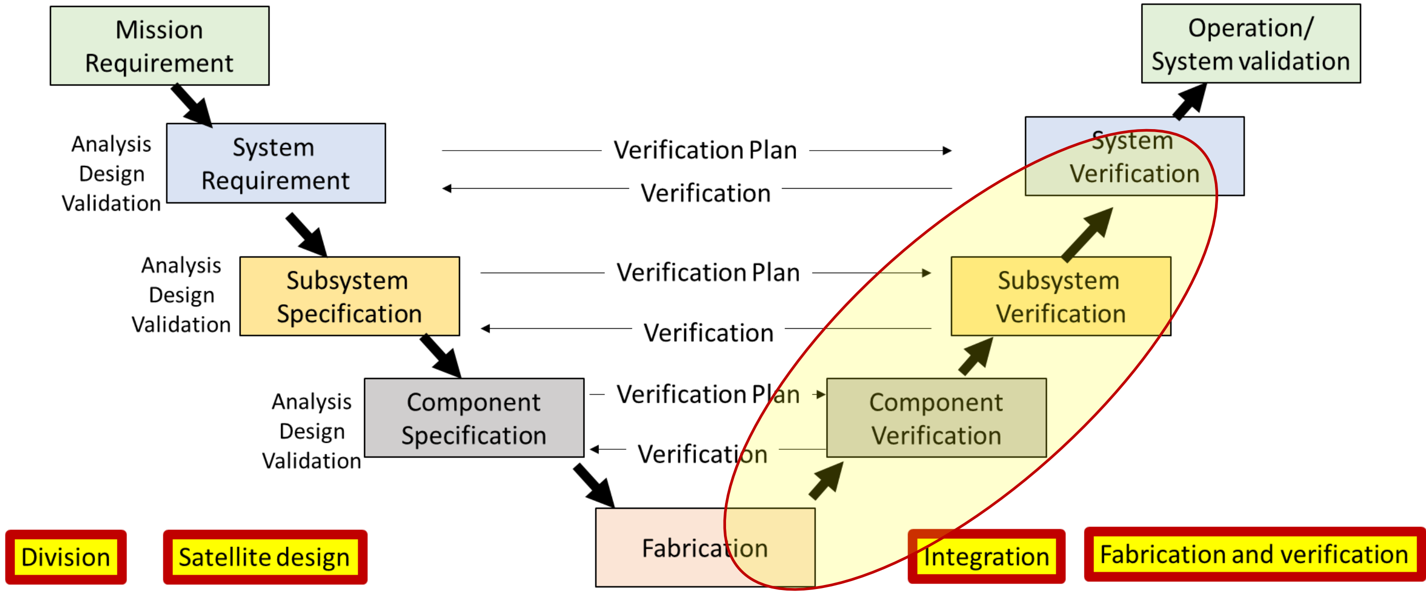
- Test model, partial models
- Development testing
- Verified functional feasibility

- Test as much as you can
- Verified whole system design
- Function/performance check
- Interface compatibility
- Environmental endurance (worst)
- Qualification test

- Verified implementation
- Environmental endurance (realistic)
- Acceptance test

1.2 Scope of this lecture

- This lecture focuses on the system integration and electrical testing
  - Verification plan and process
    - Component verification
    - Subsystem verification
    - System verification
  - Check points and related tests
  - Identify and update risks
  - Documentation (Manual, Test results, lessons learned …)



This process is also applied to software!!

1.3 Related lectures

This lecture is related to the following lectures:

- Lecture 03: Overview of Project Management of Satellite Development
  (Flow of Satellite Development and Review Meetings)
- Lecture 04: Systems Engineering for Micro/nano/pico-satellites
  (Subsystems and their relationships)
- Lecture 10: Introduction to CubeSat Command and Data Handing System
  (CubeSat C&DH Software, CubeSat System Integration, Functional Verification of C&DH System)
- Lecture 14: Introduction to CubeSat Attitude Control System
  (Functional Verification of Attitude Control System)
- Lecture 15: Introduction to Satellite Testing
  (Verification)

2. System integration

2. System integration

2.1 Overview of system integration

- System integration is performed in a bottom-up manner – easily clarify the source of errors

Component verification → Subsystem verification → System verification

HARDWARE

Component verification

- Function
- Performance
- Interface
  - standard equipment
    → actual components
  - Power, signal, mechanical, thermal …
- Environmental endurance

Subsystem verification

- Function
- Performance
- Interface
- Environmental endurance

System verification

- Function
- Performance
- End-to-end tests
  - Ground system
  - Data processing system
- Environmental endurance

SOFTWARE

Component verification

- Single function unit
  - Component driver
  - Single function

Subsystem verification

- Subsystem
  - Integration
  - Subsystem function
  - Timing/performance

System verification

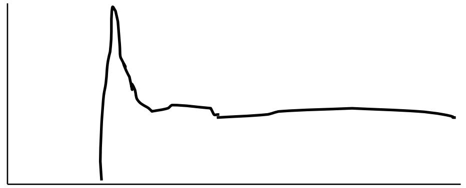
- Whole software
  - Mode transition
  - Function
  - Timing/performance

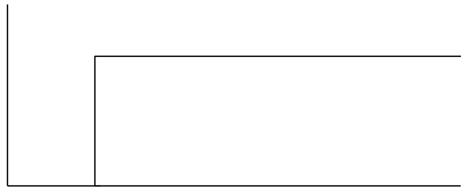
2.2 Electrical interface test (1)

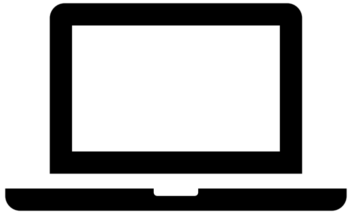
Step-by-step verification Component verification

Single unit test Compare to component specification

- Power interface
  - Pin assignment – resistance measurement
  - Voltage (lower, upper)
  - Current (nominal, rush current (transient peak))
- Signal interface
  - Pin assignment
  - Data interface (ref. lecture 10)
    - Analog (passive, active), Serial (signal lines, logic, clock, topology, protocol, voltage level …)
    - Radio frequency (frequency, modulation, (coding))
    - Data structure







- **Check the component realizes the designed interface**
- **Produce a test report**

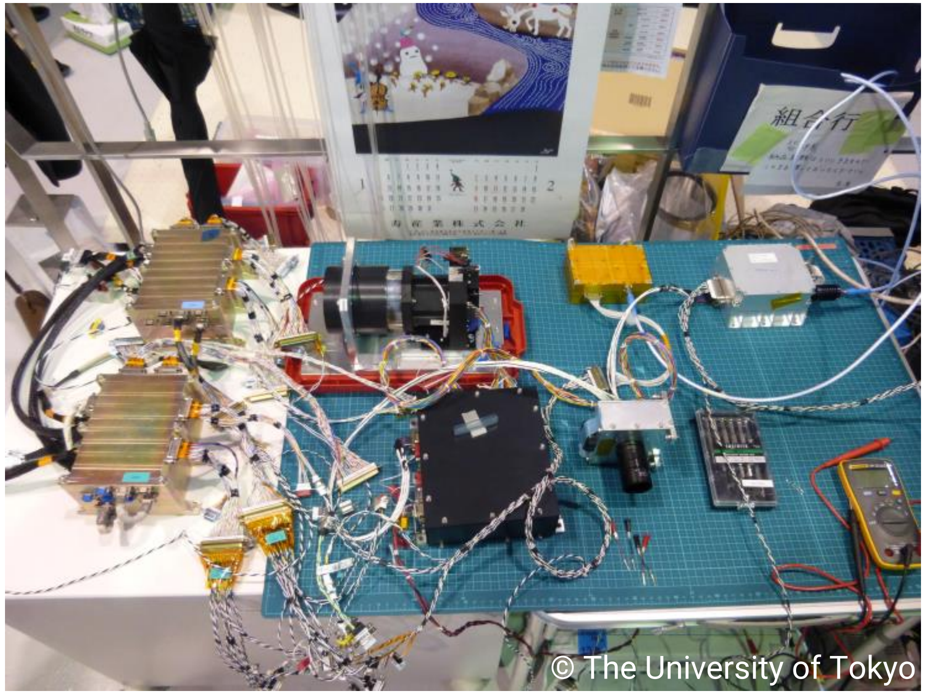
2. System integration

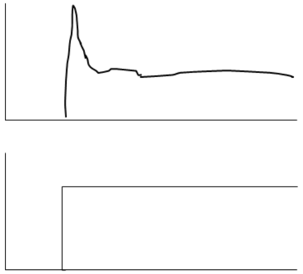
2.2 Electrical interface test (2)

Components interface compatibility test

- Check single unit test report
  → move on to the interface compatibility test

- Power interface - Power component
  - Pin assignment – resistance measurement
  - Voltage
  - Current
- Signal interface - On-board computer
  - Pin assignment
  - Data interface
  - On-board computer's software
    - Driver, debug software





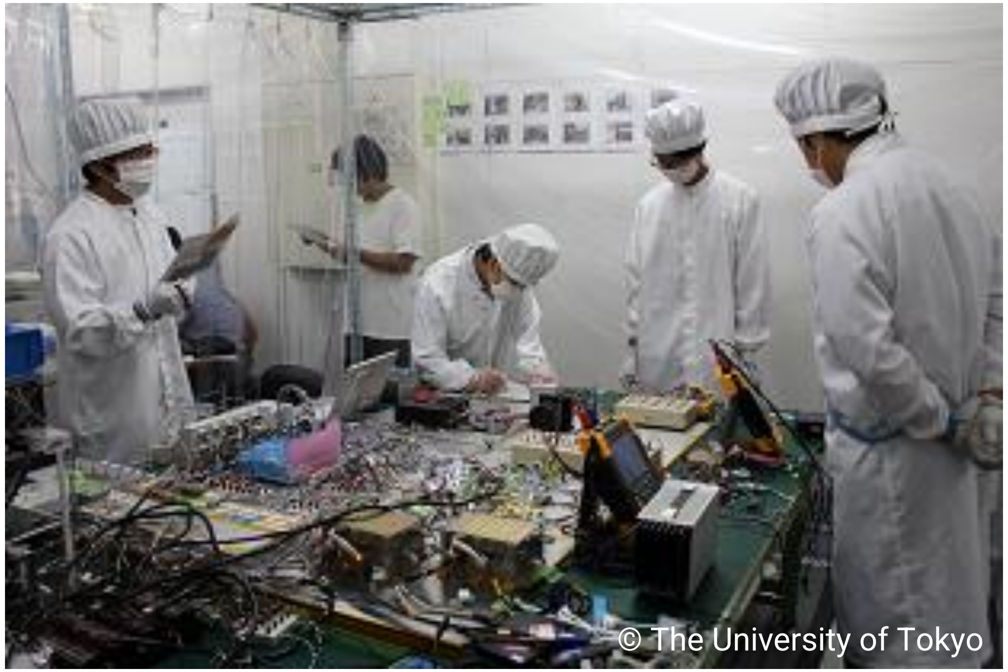
- **One by one**
  → sub-system
- **Produce a test report**

Subsystem verification

2. System integration

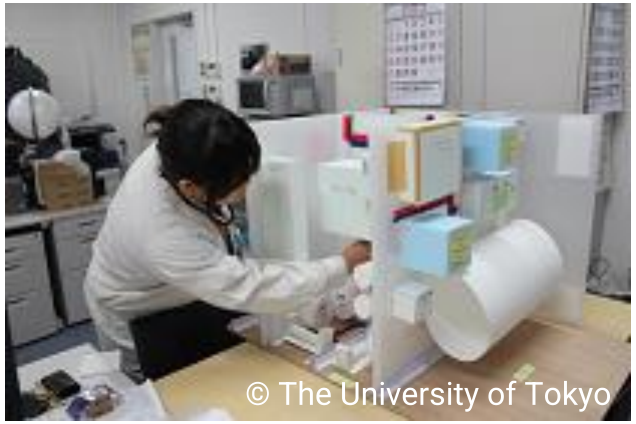
2.3 Table Sat / Functional test

- Table Sat



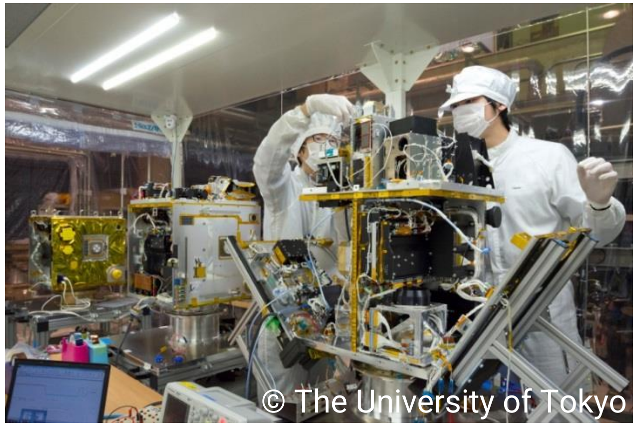
- **Check sub-system, system interface compatibility**
- **Functional test**
- **Produce a test report**
- **Compare the results with designed value and former test results**

- Mechanical layout



- Check feasibility (room for tool …)
- Harness routing
- Integration procedure

- Integration



- Check power / data interface step-by-step

2. System integration

2.3 Table Sat / Functional test

System verification

Compare to system requirements

Integrated satellite system

- Ensure the satellite works in laboratory before you conduct the environment tests
- **Whole system function test**
  - **Power on/off**
  - **Power budget**
  - **Data**
  - **Operational mode**
  - **Flight software**
  - **Produce a test report**
  - **Compare the results with designed value and former test results**

2. System integration

2.4 End to End test

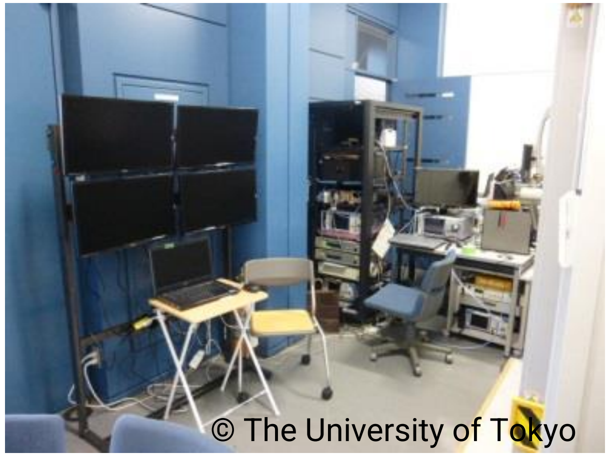
System verification

Compare to system requirements

Integrated satellite system

© The University of Tokyo

RF equipment



Ground station software

Ground station simulator

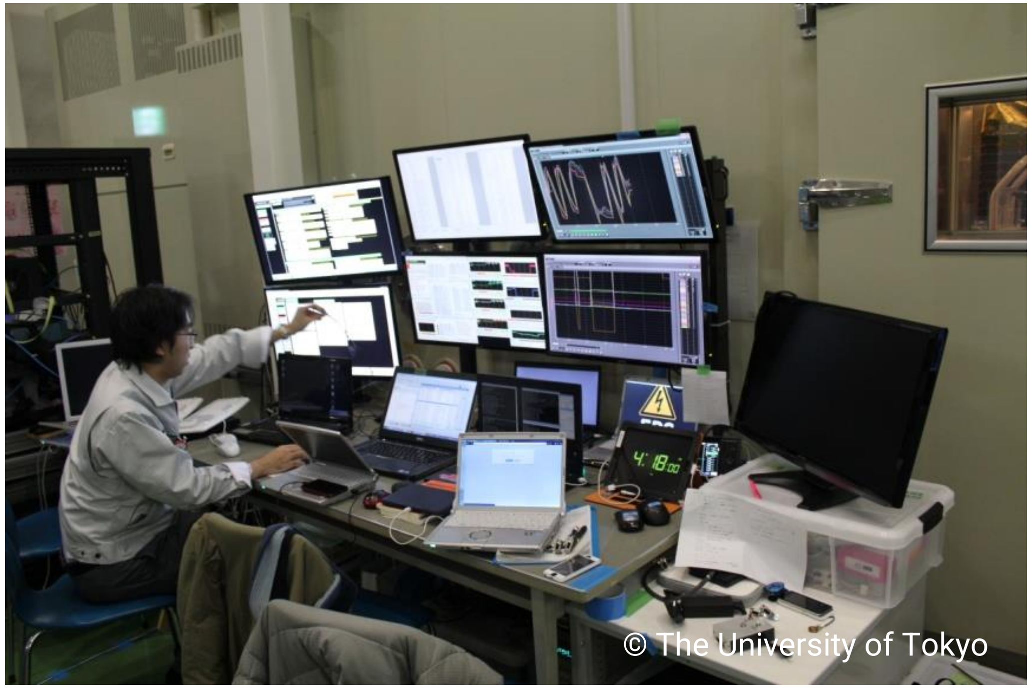
- **Flight software**
- **Ground station software, monitoring system**
- **Compare the results with designed value and former test results**

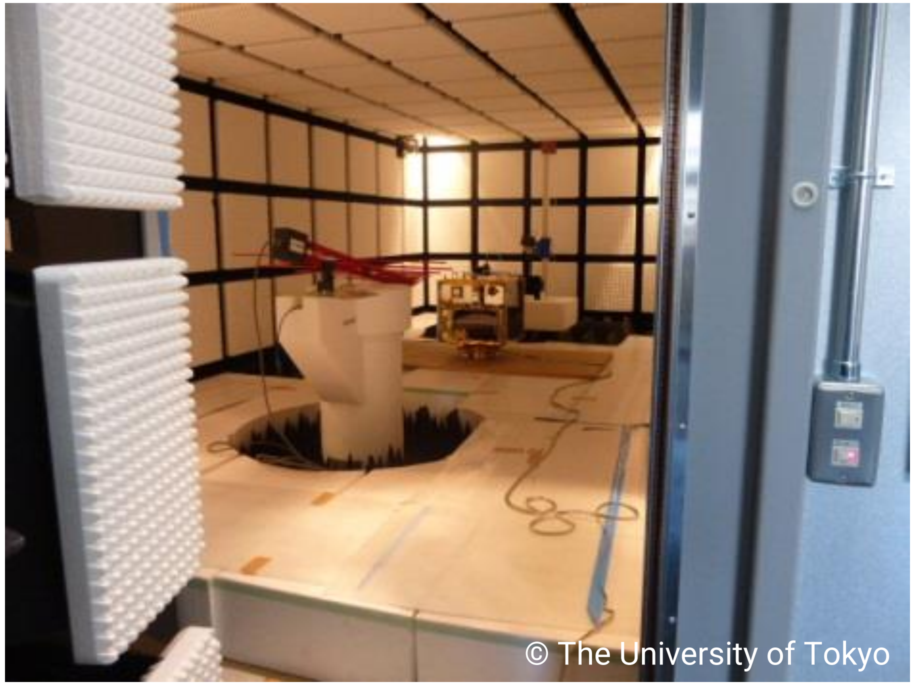
2. System integration

2.4 End-to-End test

System verification

Compare to system requirements





Check whole data flow from GS to Sat to GS

Cmd (GS) → RF (GS) → Com(RF) (Sat) → C&DH(Sat) → Action → C&DH(Sat) → Tlm (Sat) → Com(RF) (Sat) → RF (GS) → Tlm (GS)

3. Importance of software development and verification

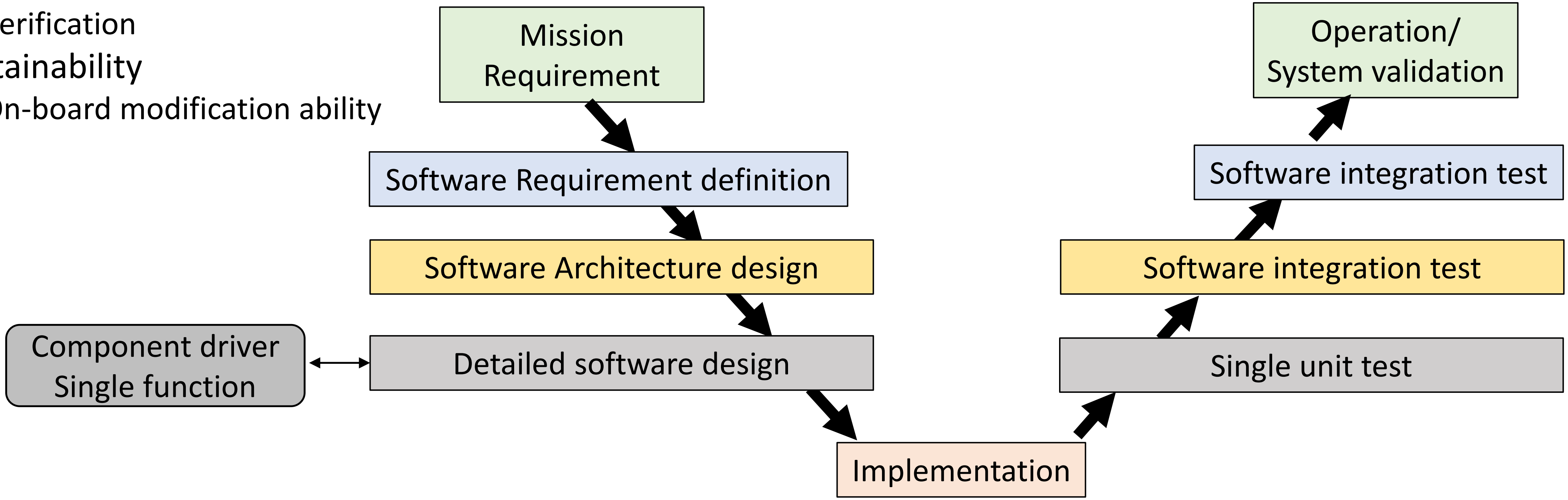
3.1 Software development

- **Design and fabricate the system achieve mission requirements**

Key points:

- Reusability
  - Divide hardware related part
- Reliability
  - Verification
- Maintainability
  - On-board modification ability

Without software, the satellite never works !
But most people focus on hardware …



3.1 Software development

- **Data packet structure (Sat ↔ Ground station)**

Header	Contents	Footer

- **Component driver**

Command interface

Command from on-board computer

Header | Contents | Footer → Component

Components interface compatibility test

Transfer command from ground

Header | Contents | Footer → Header | Contents | Footer → Component

End-to-end test

Telemetry interface

Obtaining, checking, formatting telemetry data → Transfer to ground

Component → Header | Contents | Footer →

Component → Header | Contents | Footer →

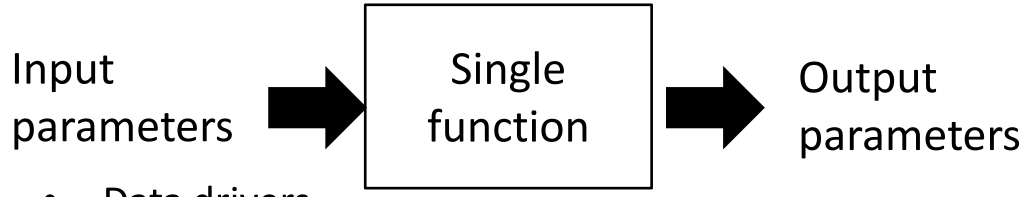
Memory → Header | Contents | Footer

Components interface compatibility test

End-to-end test

3.1 Software development

- **Single function**

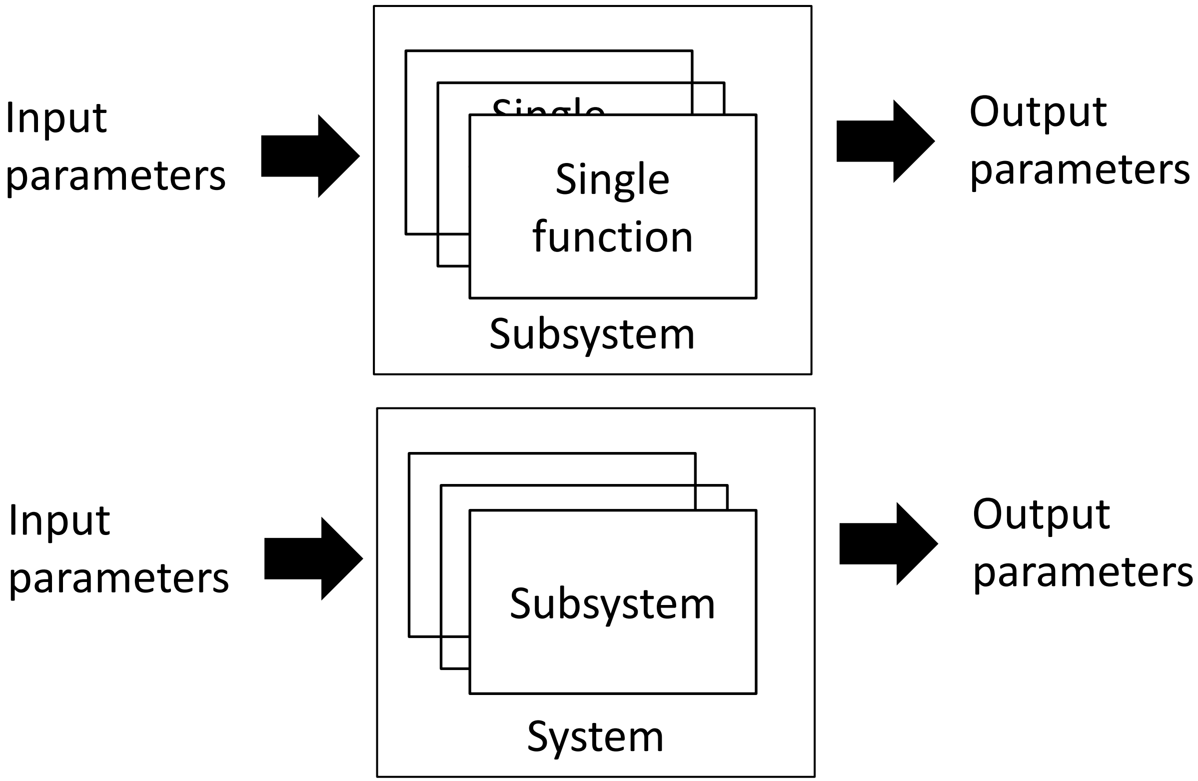


  - Data drivers
  - Output from other function

- Metaethical subroutine
- Logical subroutine …

Related component
Behavior/status

- **Subsystem, System definition**

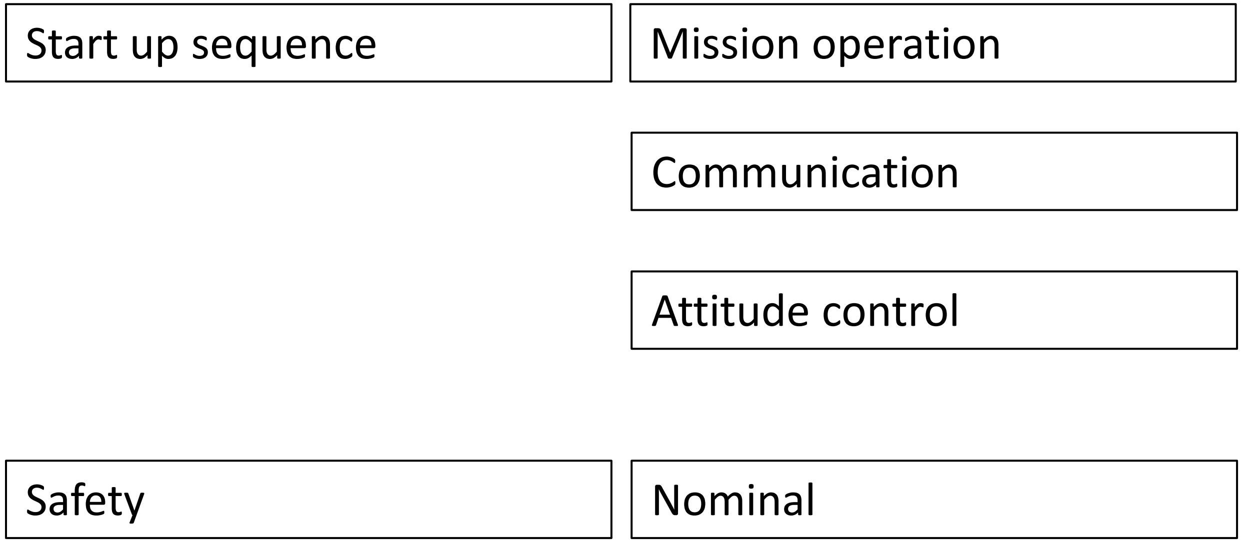


- Periodical sequence
- Event triggered

Function
Performance
Timing

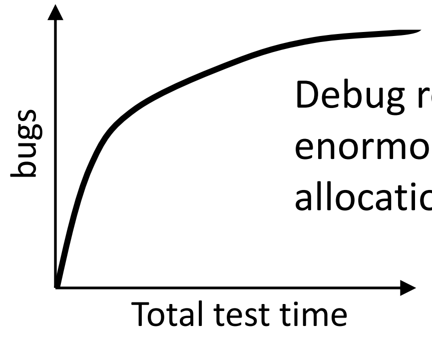
- **Mode definition**

  - Design operational mode depends on the requirements
    - Mission operation mode
    - Safety mode …
  - Mode transition definition
    - Autonomous
      - Fault detection – safety
      - Certain threshold achievement
    - By command
  - Realize the defined mode by software



3.2 Software verification

- Step-by-step verification
  - Same as the system integration steps
  - Compare to specification and designed function/performance
  - Fundamental parts are related to the components and system integration sequence



Debug port, debug software

Single unit test

Subsystem test

Components interface compatibility

Data interface

Logic

Function
Performance
Timing

System test

Related component
Behavior/status

End-to-end test

- All commands can be sent by ground station software
- All status can be seen from ground station software
- Performance / function related parameters have to be modified via ground command
- It is impossible to debug every bug in software
- Test cases have to be defined for debugging

4. Case Study

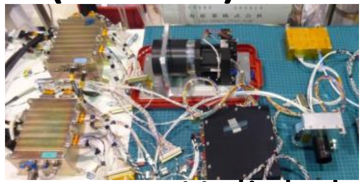
4.1 Hodoyoshi-3, 4 / UNIFORM-1

- Integration of the communication components : design / implementation verification test plan

Component test

- Status
- Command
- Voltage
- Current
- Function
- Performance
- Mechanical
- Environment

Interface (satellite)



- Status
- Command
- Uplink data
- Downlink data
- Voltage
- Current
- Mechanical

Interface (ground station)

Satellite

Data → Coding →RF

On-board computer

Data frame/packet

Error correction coding

Transmitter

Ground station

RF → Decoding →Data

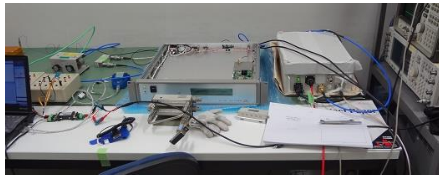
Down converter

Demodulator

On-board computer

Error correction decoding

Data frame/packet



Data format compatibility

Error coding / decoding compatibility

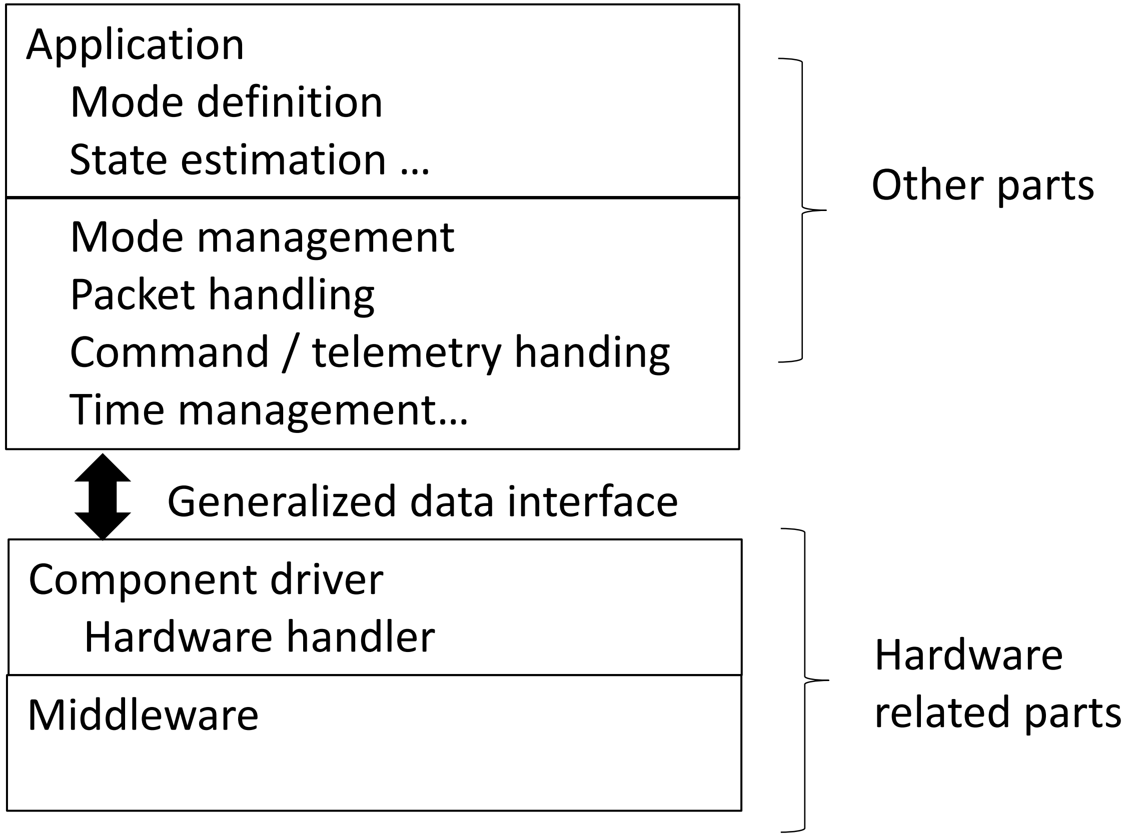
Modulation / demodulation compatibility

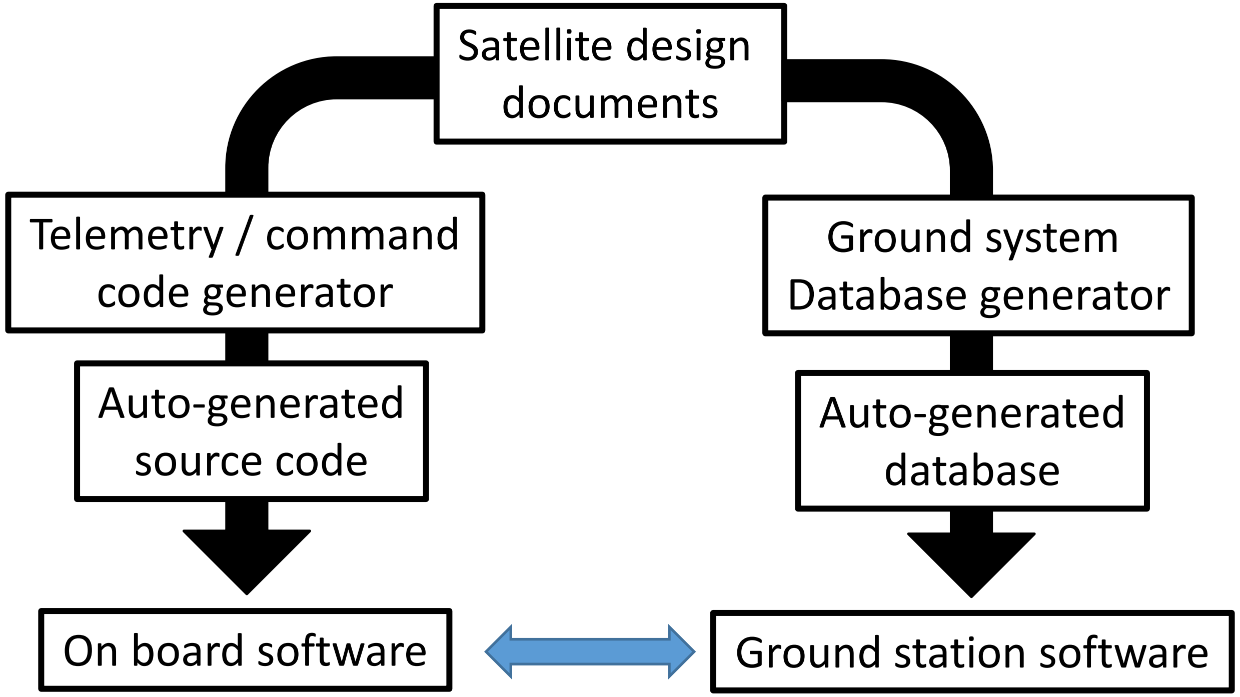
Unit test + interface test → system test

4. Case study

4.1 Hodoyoshi-3, 4 / UNIFORM-1

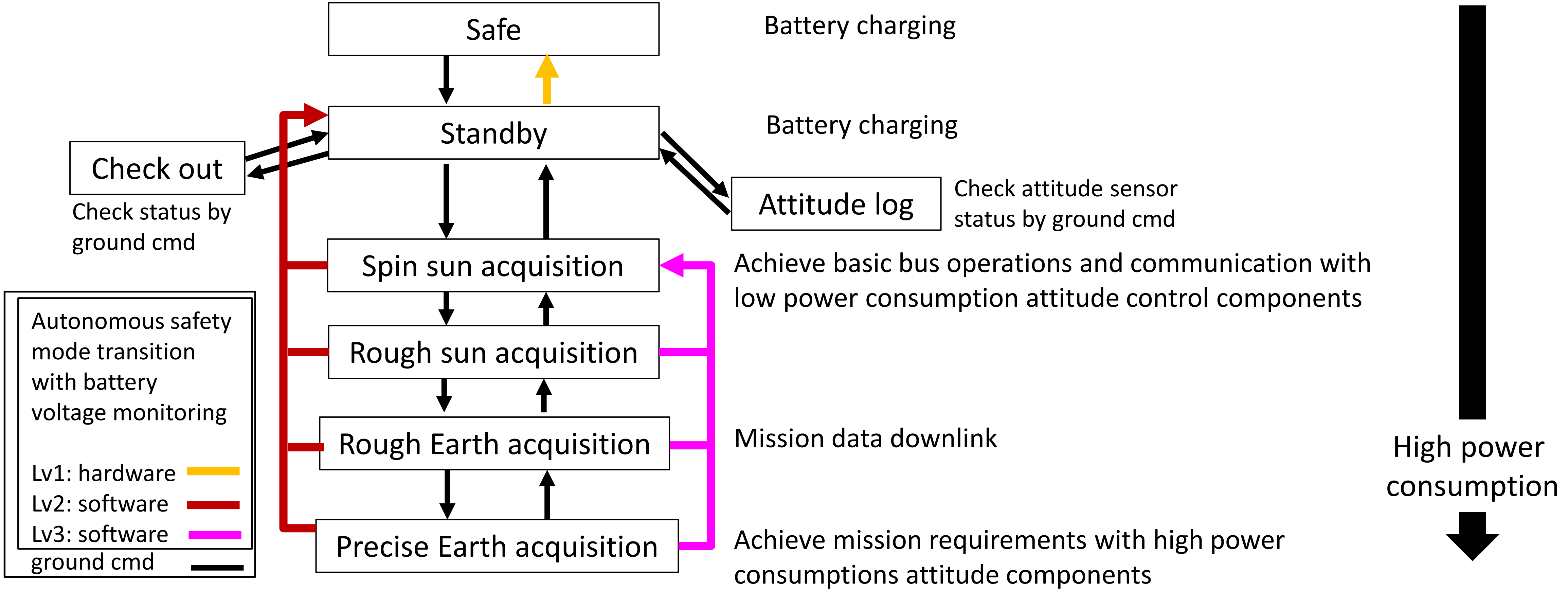
- Software design and implementation
  - Divide software functions into hardware-related parts and other parts
  - Automatic code generation system for telemetry and command related software for on-board / ground station software





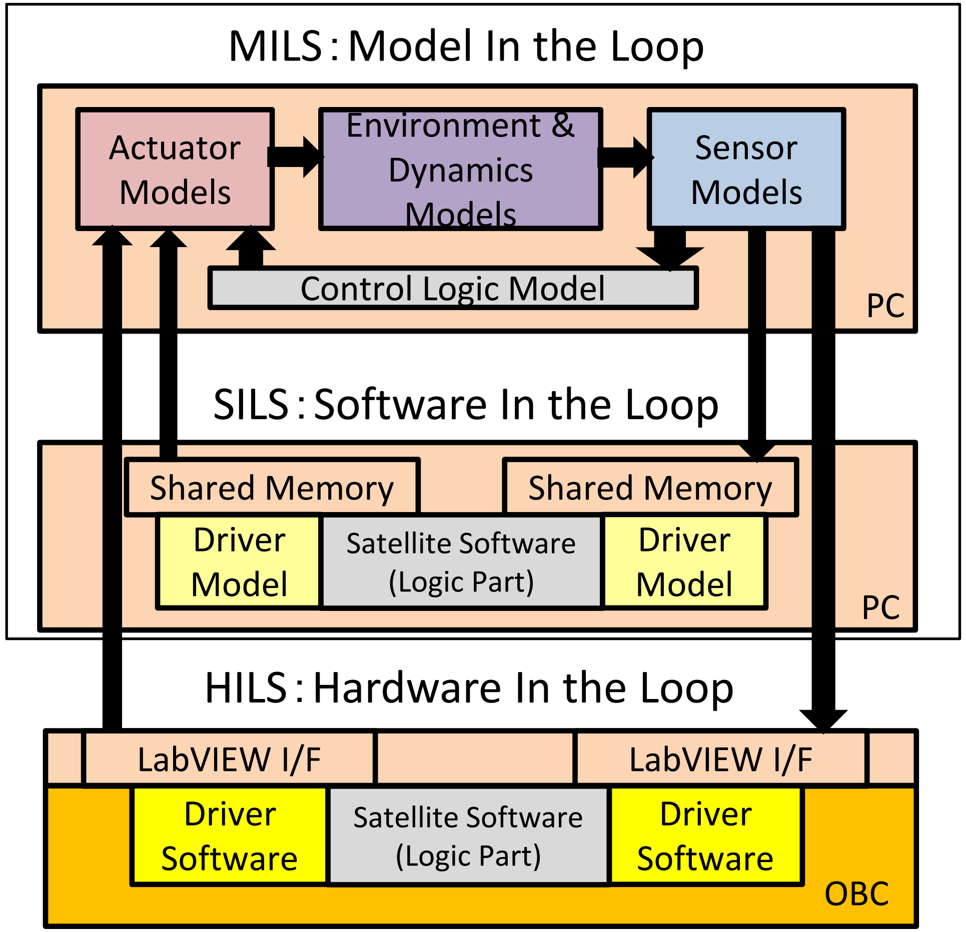
4.1 Hodoyoshi-3, 4 / UNIFORM-1

- Mode definition
  - Define modes and mode transition based on system requirements (mission achievement and survival)
  - Check feasibility with numerical simulator combined with system test results



4.1 Hodoyoshi-3, 4 / UNIFORM-1

- Software verification platform

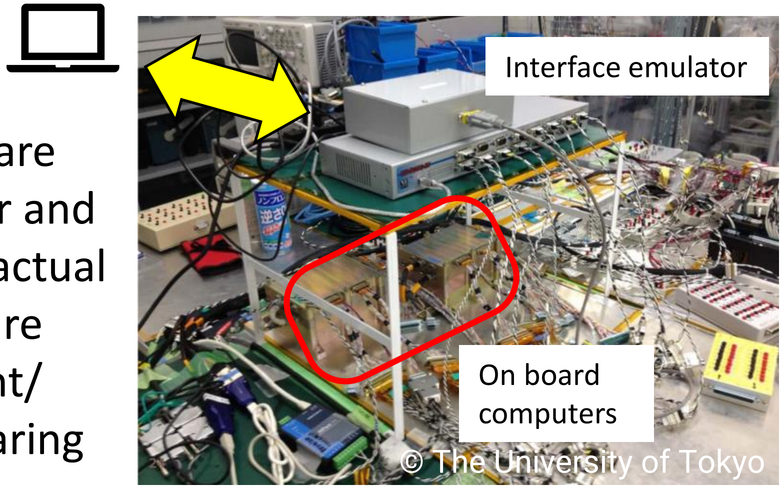


1. MILS
   Test control logic, performance, mode transition …
   with numerical simulator
   (emulate various environmental condition, check logic of the application part)

   Possible conditions are verified through numerical simulation

2. SILS
   Test implemented software
   with numerical simulator
   (check actual code (application part) comparing with MILS results)

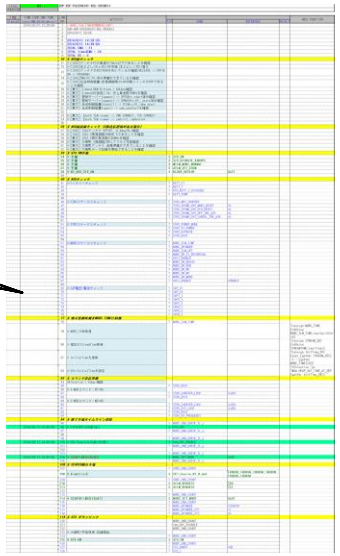
3. HILS
   Test implemented software
   with numerical simulator and actual hardware (check actual code with actual hardware performance, component/hardware drivers, comparing SILS results)



4.1 Hodoyoshi-3, 4 / UNIFORM-1

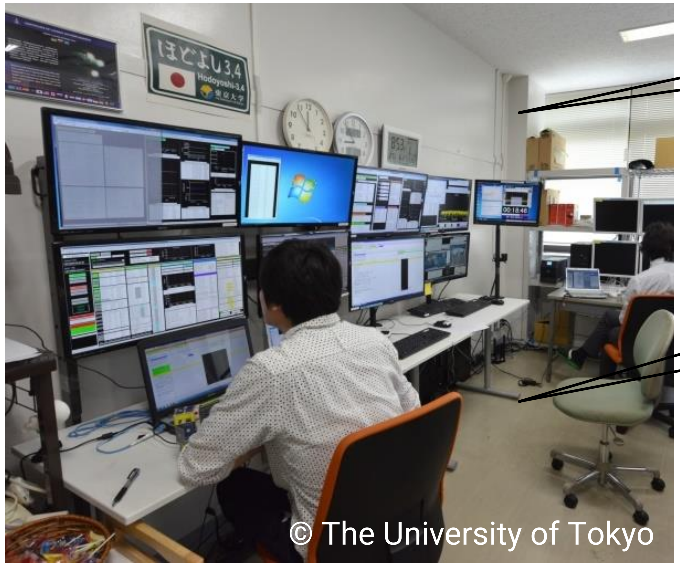
- Operational training
  - Make operation procedures for defined phases (initial operation, nominal mission operation, emergency …) and perform operation training based on the procedure
  - Operation procedure (command and check points), flight software, and ground station software also have to be verified

Satellite Operation Procedure (SOP)



Status check
Command list

Ground station



© The University of Tokyo

Check all status can be seen from ground station

Check all important parameters can be modified by command from ground station

4.2 Utilizing open source software

Software development features

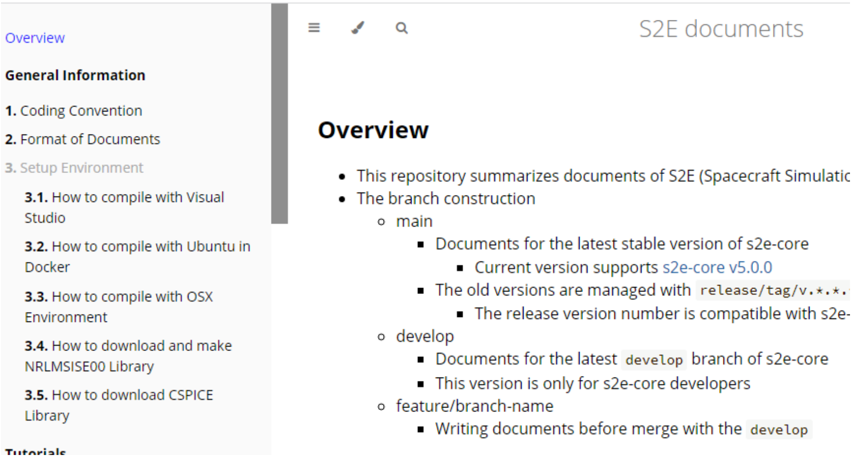
- Software testing requires test cases definition and it is very difficult for new players
- Debug process requires enormous time allocations
- Fundamental functions are required in many satellites

Utilizing open source software

Open Source Virtual Satellite (project run by ISSL, UT, member of UNISEC)

https://github.com/ut-issl

- Spacecraft on-board software architecture
  - Command Centric Architecture (C2A)
- Space environmental simulator
  - Spacecraft Simulation Environment (S2E)
- Ground station software
  - Web-based INterface Ground-station Software (wings)



5. Conclusion

5. Conclusion

- This lecture introduced the details of system integration and electrical testing.
- System integration and verification have to be performed in a **bottom-up manner** – **"step-by-step"** — **compared with the requirements and design**, which clarifies the sources of errors.
- The test results have to be **summarized in a report**, and the differences in the results of each phase have to be confirmed.
- The test results have to be confirmed by referring to the system requirements and design.
- **Both hardware and software** have to be verified during the integration procedure.
- On-board flight software has to be confirmed **together with ground software and operation procedures**.
- Utilizing **open source software** is one option to ensure reliability.

Thank you very much.

[Disclaimer]
The views and opinions expressed in this presentation are those of the authors and do not necessarily reflect those of the United Nations.