Standardization of Ground Systems Based on a Spacecraft Functional Model

Takahiro Yamada (JAXA/ISAS)

March, 2010

GSAW 2010

Agenda

- Why do we need a model?
- Functional Model of Spacecraft (FMS)
- Object Monitor and Control Protocol (OMCP)
- Spacecraft Information Base (SIB)
- Generic Spacecraft Test and Operations Software (GSTOS)
- Conclusion

WHY DO WE NEED A MODEL?

Present Design of Spacecraft

- Presently, the functional design of spacecraft onboard components is performed almost independently for each spacecraft and each component without using standards.
  - This prevents reuse of onboard components and their supporting ground systems from spacecraft to spacecraft.
- Furthermore, there are no standards for description of functional design and operational rules for onboard components. Therefore,
  - Each designer of onboard components has to devise a way of describing the functional design and operational rules for his or her components, and
  - Each operator on the ground has to learn how to read the descriptions written by the designers.

Functional Model of Spacecraft

- In order to solve these problems, we developed a method of standardizing ground systems and operations by using the Functional Model of Spacecraft (FMS).
- FMS provides a standard way of designing and describing the functions of onboard components. It is based on the object-oriented software technology, and onboard components are designed and described as Functional Objects (FOs).
- We also developed a protocol, called the Object Monitor and Control Protocol (OMCP), that specifies the formats of telecommand and telemetry messages. OMCP provides capabilities to monitor and control Functional Objects.
- The properties of the Functional Objects and the formats of telecommand and telemetry messages for each specific spacecraft are stored in a standard database called the Spacecraft Information Base (SIB).

FUNCTIONAL MODEL OF SPACECRAFT (FMS)

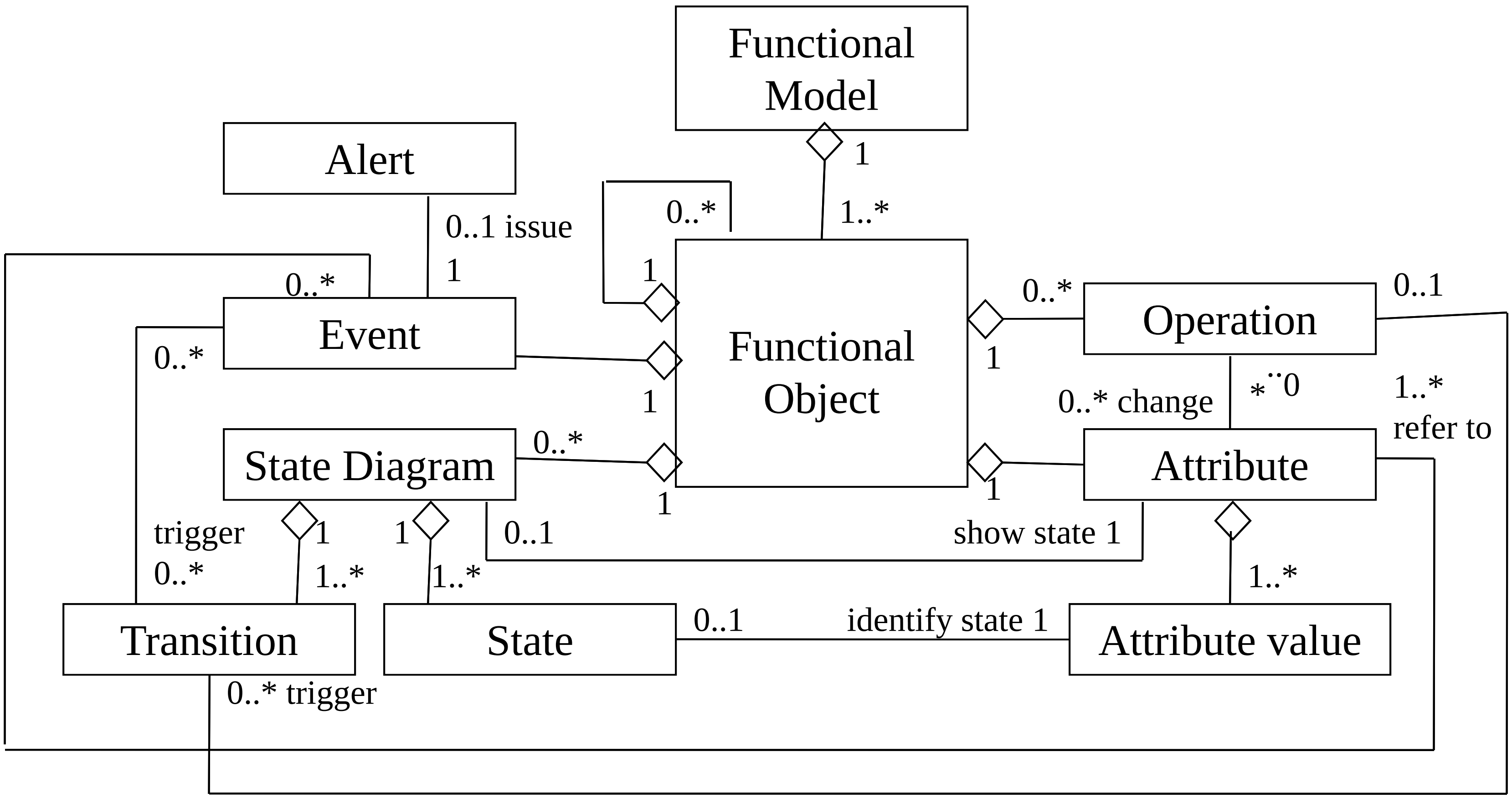
Functional Object (FO)

- The Functional Model of Spacecraft (FMS) provides a framework for designing and describing the functions of onboard components.
- In this model, a group of closely related functions of an onboard component is represented as a Functional Object (FO).
- Each Functional Object has the following properties:
  - Attributes – parameters that represent the status of the FO and can be sent to the outside world with telemetry messages;
  - Operations - functions of the FO that can be invoked by sending command messages from the outside world
  - Alerts - functions to actively report to the outside world with telemetry messages that some important event has occurred inside the FO.
  - Behavior – see next page.

Behavior of FO

- The behavior of an FO is represented with one or more state diagrams.
- State transitions are triggered either as a result of performing an operation or by the occurrence of an internal event.
- A state diagram is either active or inactive at a time.
- In what state of a state diagram the FO is in is indicated by the value of an attribute associated with that diagram.
- Each state of an FO determines the set of operations that can be invoked when the FO is in that state.
  - Therefore, a state diagram specifies the correct sequences of operations that can be performed by the FO.

UML Representation of FMS



Example: Functional Object X_A

- Attributes:
  - X_A_OnOff,
  - X_A_RunStop,
  - X_A_ErrorStatus,
  - X_A_CheckMode

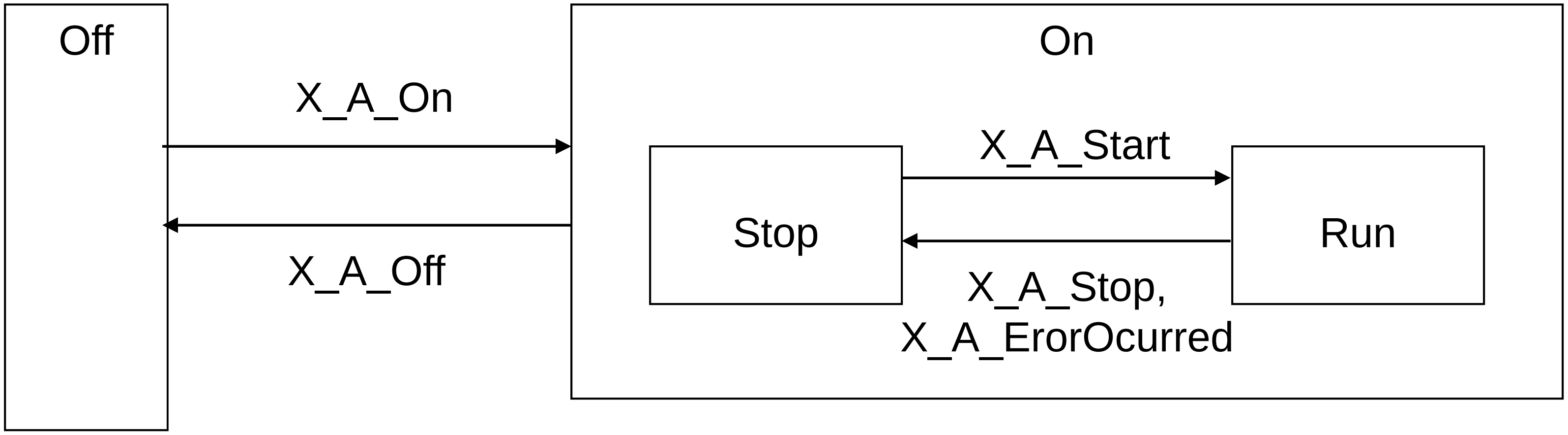
  The first two attributes show the states that this FO is in.

- Operations:
  - X_A_On,
  - X_A_Off,
  - X_A_Start,
  - X_A_Stop,
  - X_A_SetCheckMode

- Alert:
  - X_A_ErrorDetected

Behavior of X_A

- The behavior of this FO can be represented with two state diagrams.
- Since the second diagram is active only when the FO is in state On of the first diagram, it is shown contained in the On state of the first diagram.
- The labels attached to the arrows are the triggers of the transitions.
  - X_A_ErrorOccurred corresponds to an internal event.
  - The other four triggers correspond to operations.



OBJECT MONITOR AND CONTROL PROTOCOL (OMCP)

Object Monitor and Control Protocol (OMCP)

- The Object Monitor and Control Protocol (OMCP) is an application layer protocol to monitor and control Functional Objects.
- The Protocol Data Units (PDUs) of OMCP are typically transferred by the CCSDS Space Packet Protocol, but they can be transferred by any network or transport layer protocol.

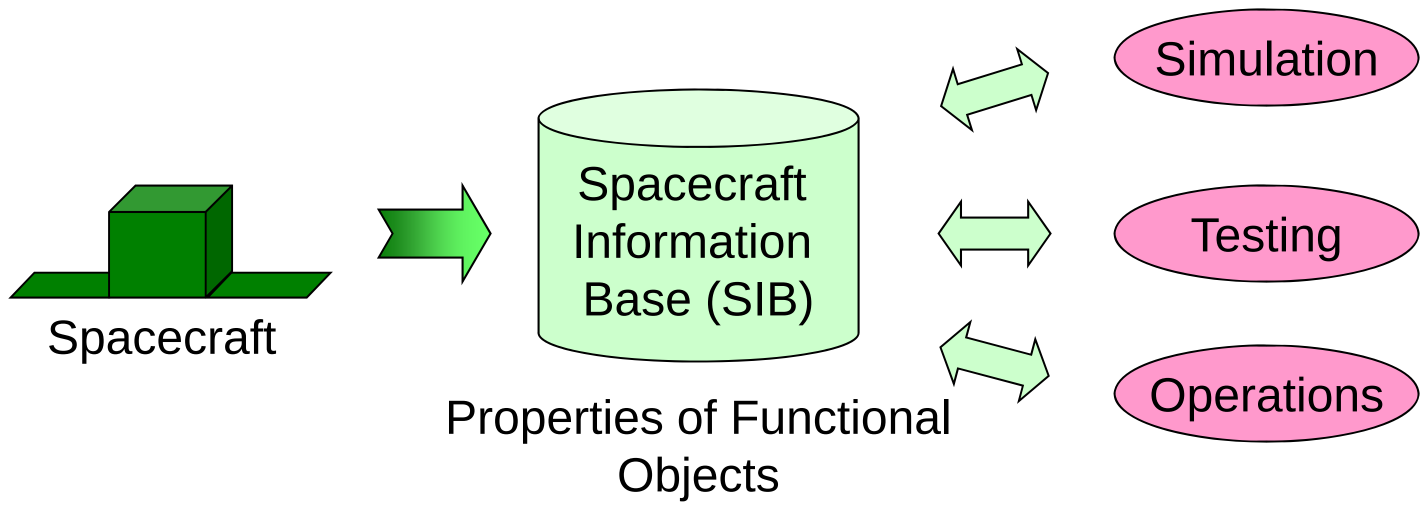
OMCP Messages

- OMCP defines types and formats of telecommand and telemetry messages and their exchange sequences.
  - Telecommand message types
    - Action Command – Invokes an operation
    - Get Command – Retrieve attribute values
    - Memory Load Command
  - Telemetry message types
    - Value Telemetry - Send attribute values
    - Notification Telemetry- Notify occurrence of an alert
    - Command Acknowledgement
    - Memory Dump Telemetry

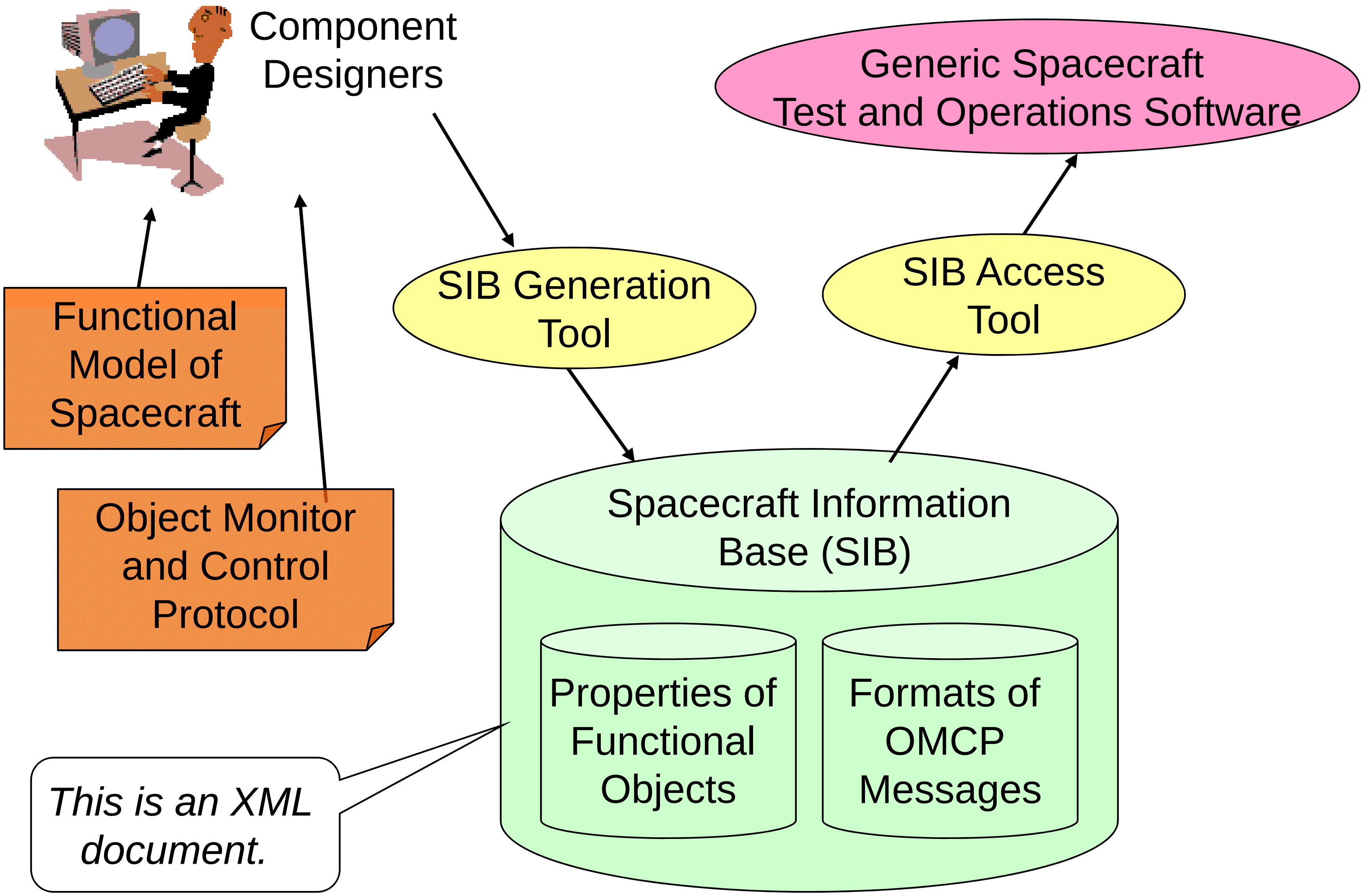
SPACECRAFT INFORMATION BASE (SIB)

Spacecraft Information Base (SIB)

- The Spacecraft Information Base (SIB) is a standard database that stores
  - The properties of Functional Objects, and
  - The formats of OMCP messages.
- The contents of the SIB can be used for simulation, testing, and operations of spacecraft.
- SIB can also be used as an online operations manual of spacecraft.



Generation and Utilization of SIB



17

Excel Based SIB Generartion Tool

functionalObject definition										kind	kind property
0	1	2	3	4	5	6	7	8	9		
Nozomi										**functionalObject**	
	ScienceInstruments									**functionalObject**	
		PWA								**functionalObject**	
			CPU_Status							**stateAttribute**	**stateMachine** stateMachineCPU_Status
			CPU_RunStatus							**stateAttribute**	**steteMachine** stateMachineCPU_RunStatus
			HTR_Status							**attribute**	Enumeration {Ena, Dis}
			ECC_Status							**attribute**	Enumeration {Ena, Dis}
			ECC_1BitErrStatus							**attribute**	Enumeration {Error, NoError}
			ECC_2BitErrStatus							**attribute**	Enumeration {Error, NoError}
			WDT_Status							**attribute**	Enumeration {ENA, DIS}
			WDT_ErrStatus							**attribute**	Enumeration {Error, NoError}
			PWA_PacketFormat							**attribute**	Enumeration {1, 2, 3, 4}
			CPU_ON							**operation**	
			CPU_OFF							**operation**	
			CPU_RUN							**operation**	
			CPU_RST							**operation**	
			HTR_ENA							**operation**	**attributeChangeRule** changeRuleHTR_StatusEna
			HTR_DIS							**operation**	**attributeChangeRule** changeRuleHTR_StatusDis
			ECC_ENA							**operation**	**attributeChangeRule** changeRuleHTR_StatusEna

Generic Spacecraft Test and Operations Software (GSTOS)

Generic Spacecraft Test and Operations Software (GSTOS)

- Using the Functional Model of Spacecraft (FMS), the Object Monitor and Control Protocol (OMCP), and the Spacecraft Information Base (SIB), generic monitor and control software can be developed to monitor and control any onboard component or any spacecraft (represented as Functional Objects).
- We are developing such software and it is called the Generic Spacecraft Test and Operations Software (GSTOS).
- The GSTOS can be used for component testing, spacecraft integration and testing, and flight operations.
- Since the behavior of any onboard component is stored in the SIB, GSTOS can verify whether each component is behaving correctly or not by comparing the values of received telemetry with the behavior descriptions stored in the SIB.

Conclusion

Conclusion

- In this paper, we have shown how ground systems and operational rules can be standardized based on the Functional Model of Spacecraft (FMS) and the Object Monitor and Control Protocol (OMCP).
- Right now, we are developing onboard components of two future spacecraft (SPRINT-A launched in 2012 and ASTRO-H launched in 2014) based on these concepts.
- Any onboard component and any spacecraft can be tested and operated with GSTOS provided that they were developed based on FMS and OMCP.
- With this method, we can greatly reduce the cost of developing ground systems and increase the reliability of space operations.