On-orbit Demonstration of Satellite Software Architecture with a Flexible Reconfiguration Capability

Shintaro Nakajima, Satoshi Ikari, Masashi Tomooka, Yoshihide Aoyanagi, Ryu Funase and Shinichi Nakasuka
(The University of Tokyo)
Introduction
Micro-spacecraft development at Univ. of Tokyo

- XI-V (2005)
- PRISM (2009)
- Nano-JASMINE
- PROCYON (2014)
- Hodoyoshi-1 (2014)
- Hodoyoshi-3, 4 (2014)
Micro-spacecraft development at Univ. of Tokyo

Since each onboard software (OBS) for each mission is developed **independently**, OBS

- needs long development term
- is less reliable

New onboard software architecture which can solve these problems is necessary.
Introduction of new OBS architecture

- New OBS architecture for small spacecraft should have both **reusability** and **on-orbit reconfigurability** to achieve reliability and to shorten development term.
Derivation of OBS Architecture
Command Centric Architecture (C2A)[1]


• Command Centric Architecture (C2A) is the software framework which realizes both reusability and flexible reconfigurability by describing ALL spacecraft’s functions as “commands”.

• C2A has been used for 3 micro-satellites and 1 interplanetary micro-spacecraft on-orbit, and 3 or more spacecraft being developed.
Common pattern of spacecraft’s function: Executing “pre-defined block of processing” at a specific timing.

Ex.1: Main loop of spacecraft. -> Periodic processing for management and control of spacecraft’s state.

- Get information from sensors.
- Analyze information from sensors.
- Analyze GS CMDs, generate TLMs.
- Calculate order for actuators.
- Send command to actuators.
Common pattern of spacecraft’s function #2

Common pattern of spacecraft’s function:
Executing “pre-defined block of processing” at a specific timing.

Ex.2: Response to accidental event.

Gas Leak

Executing a block of responses.

Pre-defined response for an event.

- Close valves.
- Close cut-off valves.
- Power off propulsion system.
- Move to safe mode.
Idea of C2A

• Spacecraft’s actions are performed specific time, and are defined as a group of specific functions.
• In other words, actions are very similar to time designated commands.

Get information from sensors.
Analyze information from sensors.
Analyze GS CMDs, generate TLMs.
Calculate order for actuators.
Send command to actuators.

T=N+1, CMD#5 [Power on telescope]
T=N+3, CMD#11 [Set params of obs.]
T=N+4, CMD#15 [Take photos]
Idea of C2A

• Spacecraft’s actions are performed **specific time**, and are defined as a group of specific functions.

• In other words, actions are very similar to **time designated commands**.

• Hence, **software architecture** which defines the behavior of satellite can be implemented **based on “command” functions**.
  
  – This “command” means not only an order from ground operator but also an order by itself.

• Also, this software architecture **standardizes and modularizes** each function.
Specification of C2A

• Spacecraft’s actions are defined as **Block Commands** (Macro Commands).

• Contents of Block Commands are described in re-writable “Definition Table.”

• The relationships between functions are also described in the table.

• Common functions are called **core functions**, and satellite-specific functions are called **applications**.
  - Applications are standardized and modularized description method for functions in C2A.
Schematic of C2A

Application
- Execution
- Class Initialization
- Class Setting
- Class
- Event Information

Application management function
- App. Definition Table
- Cmd. Definition Table
- Tlm. Definition Table

Event record function (Anomaly Logger)

Timeline command function
- Deploy
- Reference

Block command function
- Block command Definition Table

Event handling function
- Event handling Def. Table

Mode Management
- Mode transition function
- Mode transition Def. Table
- Mode processing function
- Mode processing Def. Table

Legend:
- =Necessary
- =Support
- =Def. Table
Example of C2A behavior: Main loop (Task List)

Mode processing Def. Table

<table>
<thead>
<tr>
<th>Mode ID</th>
<th>BLK ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>#3</td>
</tr>
<tr>
<td>1</td>
<td>#5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>15</td>
<td>#120</td>
</tr>
</tbody>
</table>

Block Command Definition Table

| BC#0 | ΔT=1 CMD#0 | ΔT=5 CMD#2 | ΔT=N/A N/A | ΔT=N/A N/A | ...... | ΔT=N/A N/A |
| BC#5 | ΔT=0 APP #A | ΔT=2 APP #B | ΔT=3 APP #C | ΔT=N/A N/A | ...... | ΔT=N/A N/A |

This is a command which executes APP A.

Refer

Deploy

Time

Cycle

1 block of processing

Step

Min. div of App

August 5th, 2017

Nakajima et al., SSC17-WK-05
Example of C2A behavior: Anomaly Handler

**Event recorder (Anomaly Logger)**
- Time & Event ID
- Report event time & event ID
- AH App checks recent event

**Event handling Def. Table**

<table>
<thead>
<tr>
<th>Num</th>
<th>Event ID</th>
<th>Cond</th>
<th>BLK ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(3,1)</td>
<td>Once</td>
<td>#10</td>
</tr>
<tr>
<td>1</td>
<td>(3,6)</td>
<td>Cont.</td>
<td>#15</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>100</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Block Command Definition Table**

- BC#0
  - \(\Delta T=1\)
  - CMD#0
- BC#1
  - \(\Delta T=1\)
  - CMD#5
- BC#15
  - \(\Delta T=1\)
  - CMD#5
- \(\Delta T=\text{N/A}\)
- CMD#7

**Event handling Def. Table**

- Event recorder (Anomaly Logger)
- Time & Event ID
- Report event time & event ID
- AH App checks recent event

**Block Command Definition Table**

- BC#0
  - \(\Delta T=1\)
  - CMD#0
- BC#1
  - \(\Delta T=1\)
  - CMD#5
- BC#15
  - \(\Delta T=1\)
  - CMD#5
- \(\Delta T=\text{N/A}\)
- CMD#7

**Deploy BC (T=N+\Delta T)**

- T=N+1 CMD#5
- T=N+2 CMD#7

**Time designated command is executed by an App in mode processing.**

**Cycle**
- 1 block of processing

**Step**
- Min. div of App

**1 Cycle**
- 1 Step

**August 5th, 2017**

Nakajima et al., SSC17-WK-05
Schematic of C2A (Reshown)

Application management function
- App. Definition Table
- Cmd. Definition Table
- Tlm. Definition Table

Event record function (Anomaly Logger)

Command processing
- Timeline command function
- Block command function
- Block command Definition Table

Event handling function
- Event handling Def. Table

Mode Management
- Mode transition function
- Mode transition Def. Table

Mode processing function
- Mode processing Def. Table

Time Management function

Legend: ■ = Necessary, □ = Support, ○ = Def. Table
Advantage of C2A

Flexible Reconfigurability

- **Lv. 0** Changing parameters
- **Lv. 1** Modifying block commands
  - Task List (main loop)
  - Mode transition
  - Event handling action
- **Lv. 2** Modifying definition tables
  - Mode definition table
  - Mode transition table
  - Event handling table
- **Lv. 3** Partially memory rewrite
  - Addition of new application.
- **Lv. 4** Whole memory rewrite

C2A enables reconfiguration

C2A

Recovery Capability

Workload (Risk)

Major gap
On-orbit demonstration
Univ. of Tokyo Spacecraft using C2A

Hodoyoshi-3, 4 (60kg, June, 2014)

- Hodoyoshi-3, 4 are LEO micro satellites for Earth observation.
- C2A has been developed during development of these satellites.

PROCYON (65kg, December, 2014)

- PROCYON is the interplanetary micro-spacecraft.
- C2A can be applied to the OBS of PROCYON from start of development.

Many bus components are common among these spacecraft.

We’d like to reuse some parts of OBS as much as possible by using C2A.
OBS development of PROCYON

We have completed development of OBS in 5.5 months.

Hodoyoshi-3, 4 (60kg, June, 2014)

26th May
Beginning of OBS Dev.

16th July
S/W Integ. Begins

9th September
ADCS integration

PROCYON (65kg, December, 2014)

6th November
Final OBS installation

Individual funct. dev.

FM Environ. test ▲

Integ. test ▲

ADCS func. dev.

End of Hodoyoshi’s development

Hodoyoshi-3, 4 (60kg, June, 2014)
Comparison of Hodoyoshi with PROCYON

**Comparison in core functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Telemetry</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auxiliary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time management</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>Application</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event logger</td>
<td>Same</td>
<td>Modified</td>
<td>Same</td>
</tr>
<tr>
<td>Necessary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command dispatcher</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>Mode management</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>Event handler</td>
<td>Modified</td>
<td>Same</td>
<td>Same</td>
</tr>
</tbody>
</table>
Comparison of Hodoyoshi with PROCYON

- Comparison in core functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Telemetry</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auxiliary Time</td>
<td>Same</td>
<td>Same</td>
<td>Modified</td>
</tr>
<tr>
<td>Application</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>Event logger</td>
<td>Same</td>
<td>Modified</td>
<td>Same</td>
</tr>
<tr>
<td>Necessary</td>
<td>Same</td>
<td>Same</td>
<td>Modified</td>
</tr>
<tr>
<td>Command dispatcher</td>
<td>Same</td>
<td>Same</td>
<td>Modified</td>
</tr>
<tr>
<td>Mode management</td>
<td>Same</td>
<td>Same</td>
<td>Modified</td>
</tr>
<tr>
<td>Event handler</td>
<td>Modified</td>
<td>Same</td>
<td>Same</td>
</tr>
</tbody>
</table>

No functions are added or deleted.

Changes are due to the difference of number of OBC. → Same software architecture can be applied to different spacecraft.
Comparison of Hodoyoshi with PROCYON

• Comparison of application: Many applications are added/deleted because of hardware modification.

Re-usage of Hodoyoshi’s App.
Operation of PROCYON

2015/11/8 @11,000,000km
2015/11/16 @8,000,000km
2015/11/18 @6,800,000km
2015/11/23 @6,800,000km
2015/11/29 @5,200,000km
2015/12/1 @2,900,000km

Earth from PROCYON

67P/Churyumov–Gerasimenko comet
On-orbit reconfiguration of PROCYON OBS

• We have performed many reconfigurations of PROCYON’s onboard software.
  • Lv. 0: Changing parameters [Many times]
  • Lv. 1: Modifying block commands [Many times]
    • Using combination of block commands for operation of propulsion system
  • Lv. 2: Modifying definition tables [Some times]
    • Reconfiguration of TaskList (main loop) for observation by the telescope
  • Lv. 3: Partially memory rewrite [4 times]
    • Adding FDIR function of STT
    • Adding application for angular velocity estimation using STT
    • Adding general purpose functions for anomaly detection and handling
    • Reconfiguration of HK telemetry packet
  • Lv. 4: Whole memory rewrite [None]
Example of on-orbit reconfiguration

- Adding FDIR function of STT

Main loop (Task List)

- Get information from STT.
- FDIR application
- Analyze GS CMDs, generate TLMs.
- Calculate order for actuators.
- Send command to actuators.

Overall Memory region of OBC

Onboard software memory region at launch

New application with FDIR
Conclusion
Future spacecraft using C2A

- After development of PROCYON, C2A has been implemented on some LEO satellites and one deep space CubeSat: EQUULEUS.
  - These satellites have different OBC, therefore C2A can be implemented on different platforms.
Summary

- We have developed **Command Centric Architecture (C2A)**: on-board software architecture with **flexible reconfigurability and reusability**.
  - This architecture is based on command functions, especially block commands, and definition tables.
- C2A is **used by some spacecraft on-orbit**, Hodoyoshi satellites and PROCYON.
  - The r-usability of C2A was demonstrated during development of PROCYON, reconfigurability of C2A was demonstrated by operation of PROCYON.
- C2A will be used for future spacecraft in the Univ. of Tokyo and some other groups.

Contact: nakajima@space.t.u-tokyo.ac.jp / Booth:U8 (University of Tokyo)
Back-up
Classification of command

- **Real Time Command (RTC)**
  - The command is executed at the time of receiving

    ![Diagram showing Real Time Command (RTC)]

    Command is executed ASAP.

- **TimeLine Command (TLC)**
  - The command is executed at designated time.
  - This “time” is defined as time counter of OBC.
  - Spacecraft has a list of TLCs awaiting execution.

    ![Diagram showing TimeLine Command (TLC)]

    Execution is postponed to designated time.
Classification of command

- **Single Command**
  - Executes **one** function

- **Block Command (BC): Macro Command**
  - BC is a group of some single commands (TLCs).
  - BC defines the **order** of command execution and reference time of execution based on the time of deploy.
  - To define BC is called “register,” and to execute BC is called “deploy.”
  - BC becomes some TLCs after it is deployed.
Command dispatcher

• Command means generally the order from the ground station to the satellite.
  – As a case of C2A, command means also the order from satellite itself.
  – In other words, command in C2A is a function in the program.

• Commands are classified to four types shown as a table below.

<table>
<thead>
<tr>
<th></th>
<th>Real Time</th>
<th>TimeLine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>Single and Real time</td>
<td>Single and TimeLine</td>
</tr>
<tr>
<td>Block</td>
<td>Block and Real time</td>
<td>Block and TimeLine</td>
</tr>
</tbody>
</table>
Core functions of C2A

• Necessary functions: The functions which realizes common functions among any spacecraft.
  – Command dispatcher
  – Mode processing
  – Event handling

• Auxiliary functions: The functions to realize above “Necessary functions.”
  – Time management
  – Application management
  – Event recorder
Layer structure of C2A
Advantage of C2A
~Flexible Reconfigurability~

- **Lv. 0** Changing parameter
- **Lv. 1** Modifying block command
  - Tasklist (main loop)
  - Mode transition
  - Event handling action
- **Lv. 2** Modifying definition table
  - Mode definition table
  - Mode transition table
  - Event handling table
- **Lv. 3** Partially memory rewrite
  - Addition of new application.
- **Lv. 4** Whole memory rewrite

---

**Event Handling**

- Close thruster valves
- Close cutoff valve
- Power off prop. system
- Transition to Sun-pointing mode
Advantage of C2A ~Flexible Reconfigurability~

- **Lv. 0** Changing parameter
- **Lv. 1** Modifying block command
  - Tasklist (main loop)
  - Mode transition
  - Event handling action
- **Lv. 2** Modifying definition table
  - Mode definition table
  - Mode transition table
  - Event handling table
- **Lv. 3** Partially memory rewrite
  - Addition of new application.
- **Lv. 4** Whole memory rewrite

---

Response #A
- Close thruster valves
- Close cutoff valve
- Power off prop. system
- Transition to Sun-pointing mode

Response #B
- Power off prop. system
- Transition to Safe mode

---

Gas Leak

Nakajima et al., SSC17-WK-05
Advantage of C2A
～Flexible Reconfigurability～

- **Lv. 0** Changing parameter
- **Lv. 1** Modifying block command
  - Tasklist (main loop)
  - Mode transition
  - Event handling action
- **Lv. 2** Modifying definition table
  - Mode definition table
  - Mode transition table
  - Event handling table
- **Lv. 3** Partially memory rewrite
  - Addition of new application.
- **Lv. 4** Whole memory rewrite

---

- **FOG data interpret**
- **FOG driver**
- **Low pass filter**

---

- **Onboard software memory region at launch**
- **FOG driver [interpreter]**
- **FOG Low pass filter**
- **FOG driver [modified]**
Example of on-orbit reconfiguration

- Reconfiguration of TaskList (main loop) for observation by the telescope

Get information from the telescope.

Analyze information from the telescope.

Analyze GS CMDs, generate TLMs.

Calculate order for actuators.

Send command to actuators.

Overlap processing time!

Block Command Definition Table

<table>
<thead>
<tr>
<th>BC#0</th>
<th>ΔT=1 CMD#0</th>
<th>ΔT=5 CMD#2</th>
<th>ΔT=N/A N/A</th>
<th>ΔT=N/A N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>BC#5</td>
<td>ΔT=0 App #A</td>
<td>ΔT=2 App #B</td>
<td>ΔT=3 App #C</td>
<td>ΔT=N/A N/A</td>
</tr>
</tbody>
</table>
Example of on-orbit reconfiguration

- Reconfiguration of TaskList (main loop) for observation by the telescope

Get information from the telescope.

Analyze information from the telescope.

Analyze GS CMDs, generate TLMs.

Calculate order for actuators.

Send command to actuators.

Changing the time parameter of BC.

Delaying start time of app.

Block Command Definition Table

- BC#0
  - ΔT=1 CMD#0
  - ΔT=5 CMD#2
  - ΔT=N/A N/A
- BC#5
  - ΔT=0 App #A
  - ΔT=2 App #B
  - ΔT=4 App #C
  - ΔT=N/A N/A

August 5th, 2017
Nakajima et al., SSC17-WK-05