限定なし

宇宙機の機能モデル

(Functional Model of Spacecrafts (FMS))

2020年3月31日　制定
宇宙航空研究開発機構

免責条項

ここに含まれる情報は、一般的な情報提供のみを目的としています。JAXA は、かかる情報の正確性、有用性又は適時性を含め、明示又は黙示に何ら保証するものではありません。また、JAXA は、かかる情報の利用に関連する損害について、何ら責任を負いません。

Disclaimer

The information contained herein is for general informational purposes only. JAXA makes no warranty, express or implied, including as to the accuracy, usefulness or timeliness of any information herein. JAXA will not be liable for any losses relating to the use of the information.

発行

〒305-8505 茨城県つくば市千現 2-1-1

宇宙航空研究開発機構 安全・信頼性推進部

JAXA（Japan Aerospace Exploration Agency）

本書は英語で書かれた草案を日本語に翻訳し、日本の宇宙機関 JAXA により制定された。
本標準は日本語を正とする。ただし、図表の一部で英語表記しかないものについては、それらが正本となる。文章の内容に疑問点がある場合は，日本語及び英語の双方を参照の上，JAXA 安全・信頼性推進部まで連絡をすること。

This document was originally drafted in English, then subsequently translated into Japanese and authorized by the Japanese space agency, JAXA.
The English translation is for reference purposes only, except for some tables and figures that contain English only, in which case they are the original. If there is anything ambiguous about the content of the text, please refer to both the Japanese version and the English version and contact JAXA Safety and Mission Assurance Department.

Functional Model of Spacecrafts (FMS)

GSTOS 201-1.1
Issue 1.1
31 March 2023

Japan Aerospace Exploration Agency (JAXA)

This document was originally drafted in English, then subsequently translated into Japanese and authorized by the Japanese space agency, JAXA.

The English translation is for reference purposes only, except for some tables and figures that contain English only, in which case they are the original. If there is anything ambiguous about the content of the text, please refer to both the Japanese version and the English version and contact JAXA Safety and Mission Assurance Department.

本書は英語で書かれた草案を日本語に翻訳し、日本の宇宙機関 JAXA によって制定された。

本標準は日本語を正とする。ただし、図表の一部で英語表記しかないものについては、それらが正本となる。文章の内容に不明瞭な点がある場合は，日本語及び英語の双方を参照の上、JAXA 安全・信頼性推進部まで連絡をすること。

CONTENTS // 目次

1. INTRODUCTION // はじめに

1.1. PURPOSE // 目的

The purpose of this document is to specify the method to model functions of spacecrafts and their onboard instruments. The model specified in this document serves as a guideline for the functional design of spacecrafts and their onboard instruments in the sense that the functions are designed in a way that they can be specified with this model.

本書の目的は、宇宙機（つまり、人工衛星や宇宙探査機）とその搭載機器の機能をモデル化する方法を定める事である。本書で定めるモデルは、本モデルで定める事ができるように宇宙機やその搭載機器の機能を設計するという意味において、その機能設計のガイドラインにもなっている。

This model sets a set of standardized methods to specify functions of spacecrafts and their onboard instruments and to manage electronically information of their functions. This standardized model would make systematic development of functions of spacecrafts and their onboard instruments easier and make reusing existing onboard instruments or parts of them practical. Then, the ultimate purpose is to reduce the cost of development of new spacecrafts and their onboard instruments and to enhance their reliability.

このモデルは、宇宙機やその搭載機器の機能を定めると共にその機能の情報を電子的に管理する、標準化された一群の手法を与える。この標準化されたモデルは、宇宙機やその搭載機器の機能を系統的に開発する事を容易にすると共に、既存の搭載機器やその一部の再利用を現実的なものとする。これらの究極的な目的は、新たな宇宙機やその搭載機器の開発コストを削減し、信頼性を向上する事にある。

The Generic Spacecraft Test and Operations Software (GSTOS) specified in [R2] assumes that the design of the spacecraft and its onboard instruments follows this document and the Spacecraft Monitor and Control Protocol (SMCP) [R3]. The individual specifications according to this model are supposed to be stored and managed by the Spacecraft Information Base 2 (SIB2) [R1].

[R2] が定める Generic Spacecraft Test and Operations Software (GSTOS) は、宇宙機やその搭載機器の設計が本書と Spacecraft Monitor and Control Protocol (SMCP) [R3] に従っている事を前提としている。また、本モデルに従う個々の仕様は、Spacecraft Information Base 2 (SIB2) [R1] によって蓄積、管理する事が想定される。

1.2. SCOPE // 範囲

This document specifies the Functional Model of Spacecrafts (FMS), *i.e.*, how functions of a spacecraft and its onboard instruments are specified.

本書は、Functional Model of Spacecrafts (FMS), つまり、宇宙機やその搭載機器の機能をどのように定めるかを規定する。

This document does not specify how these requirements are implemented with hardware or software.

本書は、ハードウェアやソフトウェアによるこれらの要求の具現化の方法は定めない。

1.3. APPLICABILITY // 適用先

The specifications described in this document apply to the GSTOS and the software which implements the SIB2.

本書に記す仕様は、GSTOS と、SIB2 を実装するソフトウェアとに適用する。

The specifications described in this document apply also to the spacecrafts and their onboard instruments and their ground systems for the projects that adopt the GSTOS and the SIB2.

本書に記す仕様は、GSTOS と SIB2 を採用した projects の宇宙機やその搭載機器とそれらの地上システムにも適用する。

1.4. REFERENCES // 関連文書

1.4.1. Normative References // 引用文書

None.

なし。

1.4.2. Informative References // 参考文書

[R1] JAXA, “Definition of Spacecraft Information Base 2 (DSIB2)”, GSTOS 300-1.0, JERG-2-700-TP004, March 2023.

[R2] ISAS/JAXA , “Generic Spacecraft Test and Operations Software (GSTOS) Requirement Specification”, GSTOS 400, JERG-2-700-TP003, latest issue.

[R3] JAXA, “Spacecraft Monitor and Control Protocol (SMCP)”, GSTOS 200-1.1, JERG-2-700-TP002 (NOTICE1), December 2019 (March 2023).

1.5. DOCUMENT STRUCTURE // 本書の構成

This document is organized as follows:	本書は次の通り構成する。
Chapter 1 (this chapter) states the purpose, scope, and applicability of the document, and lists the references, definitions, and notations used throughout the document.	1 章（本章）は、文書の目的、範囲及び適用先を述べると共に、本書で用いる関連文書、定義、及び表記法を示す。
Chapter 2 presents an overview of the FMS.	2 章は、FMS を概説する。
Chapter 3 specifies the Functional Object, which is the core concept of the FMS.	3 章は、FMS の中心的概念である Functional Object を定める。
Chapter 4 specifies Functional Class, which is used as a template for defining the Functional Objects.	4 章は、Functional Objects を定義するためのひな型として用いるものである Functional Class を定める。
Chapter 5 specifies the Memory Functional Class, which is a Functional Class for defining the Functional Objects that represent onboard memories.	5 章は、宇宙機搭載メモリを表現する Functional Objects を定義するための Functional Class である、Memory Functional Class を定める。
Appendix A lists all the acronyms used in this document.	Appendix A は、本書で用いる略語を示す。
Appendix B shows an example of a Functional Object.	Appendix B は、Functional Object の一例を示す。
Appendix C shows the history of terminology changes.	Appendix C は、用語の変更の履歴を示す。

1.6. DEFINITIONS AND NOTATIONS // 定義及び表記法

1.6.1. Terms defined in this document // 本書で定義される用語

None.

なし。

1.6.2. Notations // 表記法

The following notations are used throughout this document.

本書は次の表記を用いる。

A paragraph that begins with “[Example]” (or “[Example *n*]”, where *n* is a positive integer) presents an example that is aimed to help readers to understand the specification, and is not a part of the specification.

“[例]”（または “[例 *n*]”、*n* は正の整数）で始まる段落は、読者の仕様の理解を助けるための例であり、仕様の一部ではない。

A paragraph that begins with “[Note]” (or “[Note *n*]”, where *n* is a positive integer) contains an informative note that is aimed to help readers to understand the specification, and is not a part of the specification.

“[注]”（または “[注 *n*]”、*n* は正の整数）で始まる段落は、読者の仕様の理解を助けるための付加情報を記したものであり、仕様の一部ではない。

A paragraph that begins with “[Rationale]” (or “[Rationale *n*]”, where *n* is a positive integer) contains a rationale for the specification, but is not a part of the specification.

“[根拠]”（または “[根拠 *n*]”、*n* は正の整数）で始まる段落は、仕様の根拠を記したものであり、仕様の一部でない。

A rationale may be simply a phrase or a (series of) complete sentence(s). In the former case, the phrase usually starts with a lower-case letter and it should be interpreted as if it was preceded with an implicit partial sentence of “The author of this document sets the specification as such”. For example, “[Rationale] in order to distinguish clearly X and Y.” means that the author of this document sets the specification as such in order to distinguish clearly X and Y.”

1.7. VERBAL FORMS // 表現形式

The following conventions apply throughout this document:

a) the auxiliary verb '**shall**' implies mandatory conditions.

b) the auxiliary verb '**should**' implies optional but desirable conditions.

c) the auxiliary verb '**may**' implies optional conditions.

d) the auxiliary verb 'can' implies capability or ability to do something.

e) the words 'is', 'are', and 'will' imply statements of fact.

The words '**shall**', '**should**', '**may**' are highlighted in **red** and **bold** font.

本書では以下の決まりに従い記述する。

「…**こと**」「…**なければならない**」は、必須な仕様示す。

「…**べき**…」は、任意であるが推奨される仕様を示す。

「…**良い**…」は、許容される仕様を示す。

「…できる…」は、何かをする事が可能な事を示す。

他のパターンの記述は、事実を示す文である。

「…**こと**」「…**なければならない**」「…**べき**…」「…**良い**…」は、読者の仕様の理解の助けのため、**赤字・太字**で示す。

[注] 本書では、要求事項を電子的に検索しやすいように、英文の '**shall**' の訳語として、「**こと**」を使用している。逆に、'**shall**' の訳語以外では「こと」は使用せず、「事」を用いている。また、英文の '**may**' に対応する訳語として、「**良い**」という当て字を使用している。逆に、'**may**' の訳語以外で「良い」は使用していない。

「A, B, 及び C」という表記は、英文の 'A, B, and C' に対応し、「A 及び B 及び C」である事を意味する。

「A, B, または C」という表記は、英文の 'A, B, or C' に対応し、「A または B または C」である事を意味する。

英語の 'a XXX ~ the XXX' という表現に対応し、日本語は「ある XXX ~ その XXX」という表現を用いる。

When a translation into Japanese is provided, the original English version and its Japanese translation are given in the left and right sides, respectively, in principle, as in this paragraph. In some cases, *e.g.* titles of sections and captions of figures/tables, the English and Japanese versions are put in a single line separated by “ // ” in this order (“English // Japanese”) or in separate lines with no delimiter in between (“English [Line-Break] Japanese”).

日本語への翻訳が存在する場合、原則として、この段落のように、英語を左側に示し、日本語を右側に示す。また、章や図表のタイトル等は、英語、日本語の順に一行中に // で区切る（「英語 // 日本語」）か、二行に分けて区切り文字なし（「英語 [改行] 日本語」）で、記述する場合もある。

In most cases, technical terms are not translated into Japanese. The English words in alphabet remain as they are in their Japanese translation. The forms in alphabet in English which distinguish the singular and plural words remain as they are in the Japanese version to preserve the information of the quantity, although the Japanese language does not inherently distinguish the singular and plural forms.

多くの場合、技術用語の翻訳は行わず、英単語を維持する。そこで、日本語にもアルファベットが登場する。それらは正本である日本語文中においてもアルファベット表記される。日本語の名詞に単数形・複数形の区別はないが、単複の情報を保つため、日本語文中においても、英語の単数形・複数形の違いはアルファベットでそのまま表記する。

Technical terms are highlighted in green and in some cases in blue. The latter consists of names of documents and protocols, widely used technical terms, and those locally used in some sections (*e.g.* field names). Note that the first character of an English word in a technical term is written in a capital letter, except for that in widely used technical terms.

技術用語は読者の便のため緑字、場合により青字で示す。後者は、文書名、プロトコル名、広く用いられている技術用語、及び、本書中の一部の章にしか登場しないもの（フィールド名等）からなる。ここで、技術用語は、広く用いられているものを除き、基本的に大文字始まりの英単語で表記する。

2. OVERVIEW

2.1. GENERAL // 一般

A model is a framework which represents something from a certain point of view. This document specifies a framework (hereafter referred to as the Functional Model of Spacecrafts, FMS) which represents a spacecraft and its onboard instruments from a functional point of view. This chapter provides an informative overview of the FMS specified in the subsequent chapters.

モデルとは、ある観点から何かを表現するためのフレームワークの事である。本書は、機能的な観点から宇宙機やその搭載機器を表現するためのフレームワーク（以下、Functional Model of Spacecrafts, FMS と称する）を定める。本章は、引き続く章で定める FMS の概要を示す。

2.2. PURPOSE OF THIS MODEL // このモデルの目的

A function of a spacecraft or onboard instrument in this model is an abstract representation of a job of the spacecraft or onboard instrument in orbit, such as performing an observation or an experiment. This abstraction is made by focusing on the outcome that outside observers can see when or after a spacecraft or onboard instrument has performed a job, disregarding how the job is performed inside the spacecraft or onboard instrument.

本モデルにおいて、宇宙機または搭載機器の機能とは、宇宙機または搭載機器が軌道上で実施する観測や実験の実施等の仕事を抽象化して表したものである。この抽象化では、宇宙機または搭載機器が仕事を実行したときまたは後に外部のオブザーバが見る事ができる結果に注目し、その仕事を宇宙機または搭載機器内部でどのように実施するかは無視する。

The FMS specifies the functions of a spacecraft or onboard instrument. If its functions are designed in a way that they follow the specifications of this model, a part of the methods for functional designs of different spacecrafts and their onboard instruments will be unified. Therefore, this model also serves as a guideline for the functional design of spacecrafts and their onboard instruments. In other words, this model also aims at standardizing the functional design of spacecrafts and their onboard instruments in terms of how a spacecraft or onboard instrument is operated by observers outside the spacecraft or onboard instrument.

FMS は、ある宇宙機または搭載機器の機能を定める。このモデルに従い機能を定める事ができるように宇宙機またはその搭載機器の機能を設計すれば、異なる宇宙機やその搭載機器の機能設計の方法の一部が統一される事にもなる。従って、このモデルは、宇宙機やその搭載機器の機能設計に対するガイドラインにもなっている。言い換えれば、本モデルは、宇宙機やその搭載機器の機能設計を、宇宙機または搭載機器をその外部のオブザーバからの運用性という観点から標準化する事を目指している。

2.3. FUNCTIONAL OBJECT

Since functions that a spacecraft or onboard instrument has are generally complex, they are usually specified as groups of functions according to their characteristics. The model specified in this document refers to an entity which has a group of functions of these kinds as a Functional Object (see Chapter 3). The Functional Object is the core concept in the FMS.

一般に宇宙機または搭載機器の機能は複雑であるため、それらはその特性に応じて機能のグループとして通常定める。本書が定めるモデルでは、このような機能のグループを持つ構成要素を Functional Object と称する（3 章参照）。Functional Object は、FMS における核となる概念である。

An entire spacecraft is a Functional Object. Any group of functions of arbitrary granularity such as subsystems, onboard instruments, and onboard software functions are regarded as Functional Objects depending upon the design of the spacecraft.

宇宙機全体は、Functional Object である。また、宇宙機の設計に応じ、そのサブシステム、搭載機器、搭載ソフトウェアの持つ機能など任意の粒度の機能の塊が Functional Object とみなされる。

A Functional Object specifies how a particular job of a spacecraft or onboard instrument appears from a point of view of observers. The concept of the Functional Object is based on the concept of the object used in object-oriented programming. In object-oriented programming, a unit of a program is defined as a class. During execution of the program, an instance (which is also called an object) is dynamically generated according to the definition of the class, and the instance executes the program. In the case of spacecrafts and their onboard instruments, the concept of object-oriented programming cannot be directly applied because a spacecraft or onboard instrument is not software but hardware (although software is used in some parts of the spacecraft or onboard instrument). However, the core concept is still applicable. In this model of the Functional Object, the functions of a spacecraft or onboard instrument are assumed to exist permanently (although whether each function is executable or not at a particular time depends on the status of the spacecraft or onboard instrument at the time). In summary, a Functional Object corresponds to an instance in the software terminology, but exists permanently unlike the counterpart in software.

Functional Object は、宇宙機または搭載機器のある特定の仕事がオブザーバにどのように見えるかを定める。Functional Object の概念は、オブジェクト指向プログラミングで用いるオブジェクトの概念に基づいている。オブジェクト指向プログラミングでは、プログラムの単位をクラスとして定義する。プログラムの実行時にはクラス定義に従いインスタンス（オブジェクトともいう）が動的に生成され、そのインスタンスがプログラムを実行する。宇宙機やその搭載機器の場合は、宇宙機または搭載機器はソフトウェアでなく（宇宙機またはその搭載機器の一部分でソフトウェアを用いるにせよ）ハードウェアであるため、オブジェクト指向プログラミングの考えを直接適用はできない。しかし、中核概念は適用可能である。Functional Object に関する本モデルでは、宇宙機または搭載機器の機能は永続的に存在するものと仮定する（ただし、ある特定の時点においてそれぞれの機能が実行可能か否かは、その時点での宇宙機または搭載機器の状態に依存する）。要約すると、Functional Object はソフトウェアの用語のインスタンスに対応するが、インスタンスとは異なり永続的に存在する。

If two or more instruments that perform the same job, such as actuators with identical specifications, are installed on a spacecraft, they are defined as two distinct Functional Objects. In this case, the definition of these multiple Functional Objects can be specified as a template for generating these Functional Objects. A template for generating multiple Functional Objects is referred to as a Functional Class (see Section 4).

同一仕様のアクチュエータ等、宇宙機に同じ仕事を行う機器が二つ以上搭載されている場合、それらは二つの別個な Functional Objects として定義される。その場合、これら複数の Functional Objects の定義は、これらの Functional Objects を生成するひな型として規定できる。複数の Functional Objects を生成するためのひな型の事を Functional Class と称する（4 項参照）。

Functional Objects can be monitored and controlled by other entities, *i.e.*, entities outside the spacecraft (for example, spacecraft control systems and spacecraft test systems) and/or other Functional Objects on the spacecraft. This document does not specify methods of communications between Functional Objects and other entities (see also Section 2.5).

Functional Objects は、他の構成要素、つまり、宇宙機外部の構成要素（例えば宇宙機管制システムや宇宙機試験システム）、宇宙機上の他の Functional Objects の一方か双方かで監視制御できる。本書は Functional Objects と他の構成要素の間の通信手段は定めない（2.5 項も参照）。

Some entities monitor Functional Objects but do not control them. An entity that monitors and controls Functional Objects is called a Controller (see Section 3.1).

一部の構成要素は Functional Objects を監視するが制御は行わない。Functional Objects を監視及び制御する構成要素は Controller と呼ばれる（3.1 項参照）。

The concepts used to specify a Functional Object include the Attribute, Operation, Event class, State Machine, and Diagnostic Rule (see Sections 3.3, 3.4, 3.5, 3.6, and 3.7, respectively).

Functional Object を定めるために用いる概念には、Attribute, Operation, Event class, State Machine, 及び Diagnostic Rule（それぞれ、3.3 項、3.4 項、3.5 項、3.6,項、及び 3.7 項参照）が含まれる。

An Attribute of a Functional Object is a variable representing a status of the Functional Object. The values of Attributes of a Functional Object at a given time can be obtained by other entities using telemetry messages.

Functional Object の Attribute は、その Functional Object の状態を表す変数である。ある時点における Functional Object の Attributes の値は、telemetry messages を用い、他の構成要素によって取得できる。

An Operation of a Functional Object is an action performed by the Functional Object and is invoked from Controllers using a telecommand message. The values of some Attributes of a Functional Object can be set with Operations invoked by Controllers.

Functional Object の Operation は、その Functional Object が行う動作であり、telecommand messages を用い、Controllers によって呼び出される。Functional Object の Attributes の一部は、Controllers が呼び出した Operations により値を設定できる。

An Event class of a Functional Object is a classification of an event (an occurrence of a thing that has a particular significance). Events of some Event classes are detected by the Functional Object itself. The event that is important to other entities can be reported to them using telemetry messages. The report is referred to as an alert and its classification is referred to as an Alert class.

Functional Object の Event class は、event（特定の意味を有する出来事の発生）の分類である。一部の Event classes の events は、その Functional Object 自体で検出する。他の構成要素にとり重要な event を、それらに、telemetry messages を用い、通知する事ができる。この通知を alert と称し、その分類を Alert class と称する。

A State Machine of a Functional Object specifies how it behaves with a finite number of states and transitions between them (see Section 3.6).

Functional Object の State Machine は、有限個の状態とその間の遷移によって、どのように動作するかを定めるものである（3.6 項参照）。

A Diagnostic rule of a Functional Object is a rule used by other entities to diagnose whether the Functional Object is functioning correctly or not.

Functional Object の Diagnostic Rule は、その Functional Object が正常に機能しているか否かを診断するための規則であり、他の構成要素が用いる。

2.4. SPACECRAFT INFORMATION BASE 2

The definitions of the Functional Objects specified with the methods specified in this document can be registered in the Function Definition Part of the Spacecraft Information Base 2 (SIB2) [R1]. The SIB2 is a database to manage information about a spacecraft or onboard instrument electronically. The information about the functions that a spacecraft or onboard instrument has **should** be managed uniformly, using this database, throughout all the phases concerning the spacecraft or onboard instrument, including designing, testing, and flight operations.

本書で定めた方法に従い定めた Functional Objects の定義は、Spacecraft Information Base 2 (SIB2) [R1] の機能定義部に登録する事ができる。SIB2 は、宇宙機または搭載機器に関する情報を電子的に管理するためのデータベースである。宇宙機または搭載機器が持つ機能に関する情報は、設計、試験、飛行運用を含むその宇宙機または搭載機器に関わる全てのフェーズを通じてこのデータベースで一元管理す**べき**である。

2.5. COMMUNICATIONS BETWEEN FUNCTIONAL OBJECTS AND OTHER ENTITIES
FUNCTIONAL OBJECTS と他の構成要素の間の通信

A Functional Object communicates with other entities. A signal for communication to a Functional Object is referred to as a telecommand message and that from a Functional Object is referred to as a telemetry message in this model.

Functional Object は他の構成要素と通信を行う。本モデルでは、ある Functional Object への通信に用いる信号を telecommand message, ある Functional Object からの通信に用いる信号を telemetry message と称する。

Although this document does not specify the methods for communications between a Functional Object and other entities, it is recommended that the Spacecraft Monitor and Control Protocol (SMCP) [R3] is used to perform the following operations (the types of telecommands / telemetries in [R3] are shown in parentheses):

本書は、ある Functional Object と他の構成要素の間の通信方法を定めないが、Spacecraft Monitor and Control Protocol (SMCP) [R3] を用いて以下の操作を実行する事が推奨される。なお、以下の箇条書きでは、カッコ内に [R3] におけるテレコマンド・テレメトリの種別を示す。

a) to invoke Operations of Functional Objects and to set the values of Attributes of the Functional Objects from a Controller (ACTION Telecommand or SET Telecommand),

a) Functional Objects の Operations の、Controller による呼び出し及び Attributes の値の設定 (ACTION Telecommand または SET Telecommand)

b) to report the values of Attributes of Functional Objects to other entities (VALUE Telemetry and NOTIFICATION Telemetry),

b) Functional Objects の Attributes の値の、他の構成要素への通知 (VALUE Telemetry 及び NOTIFICATION Telemetry)

c) to request reporting the values of Attributes of Functional Objects from Controllers (GET Telecommand),

c) Functional の Attributes の値の通知の、Controllers からの要求 (GET Telecommand)

d) to report an alert of an Alert class to other entities (NOTIFICATION Telemetry),

d) Alert class の alert の、他の構成要素への通知 (NOTIFICATION Telemetry)

e) to request uploading and dumping memory data of Memory Functional Objects from Controllers (MEMORY UPLOAD Telecommand and MEMORY DUMP Telecommand, respectively), and

e) Memory Functional のメモリデータのアップロード及びダンプ（読み出し）の、Controllers からの要求 (それぞれ、MEMORY UPLOAD Telecommand 及び MEMORY DUMP Telecommand)

f) to report memory data of Memory Functional Objects to other entities (MEMORY DUMP Telemetry).

f) Memory Functional のメモリデータの、他の構成要素への通知 (MEMORY DUMP Telemetry)

The types of telecommands/telemetries in [R3] are mentioned in this document in the sections described in Table 2-1.

[R3] におけるテレコマンド・テレメトリ各の種別は、本書において、Table 2-1 が記す箇所で言及されている。

Table 2-1: Type of Telecommands/Telemetries in [R3]
[R3]におけるテレコマンド・テレメトリの種別

Type	Section in this document
ACTION Telecommand	3.3.1 (Attributes), 3.4.1 (Operations)
SET Telecommand	3.3.1 (Attributes), 3.4.1 (Operations)
GET Telecommand	3.3.1 (Attributes)
VALUE Telemetry	3.3.1 (Attributes)
NOTIFICATION Telemetry	3.3.1 (Attributes), 3.5.2.1 (Alert classes)
MEMORY LOAD Telecommand	5.3.2 (Memory Functional Class, MemoryLoad)
MEMORY DUMP Telecommand	5.3.3 (Memory Functional Class, MemoryDump)
MEMORY DUMP Telemetry	5.3.3 (Memory Functional Class, MemoryDump)

3. METHOD OF MODELING WITH FUNCTIONAL OBJECTS
FUNCTIONAL OBJECTS によるモデル化の方法

3.1. GENERAL // 一般

Functions of a spacecraft or onboard instrument **shall** be specified with entities each of which is referred to as a Functional Object. A Functional Object has a group of functions that are easy to be specified and understood. The Functional Object is an abstract representation of a spacecraft or onboard instrument function to express solely how they appear to outside entities.

宇宙機または搭載機器の機能を、それぞれ Functional Object と称する構成要素を用いて定めること。Functional Object は、定めるのも理解するのも容易になるようにグループ化された機能を持つ。Functional Object は、外部の構成要素からどのように見えるかのみを表す、宇宙機または搭載機器の機能を抽象化した表現である。

The concepts used to specify how a Functional Object behaves are summarized in Figure 3-1, which include the Attribute, Operation, Event class, State Machine, and Diagnostic Rule (see Sections 3.3, 3.4, 3.5, 3.6, and 3.7, respectively).

Functional Object がどのように動作するかを定めるのに用いる概念のサマリを Figure 3-1 に示す。これらには、Attribute、Operation、Event class、State Machine 及び Diagnostic Rule（それぞれ、3.3 項、3.4 項、3.5 項、3.6 項、及び 3.7 項参照）が含まれる。

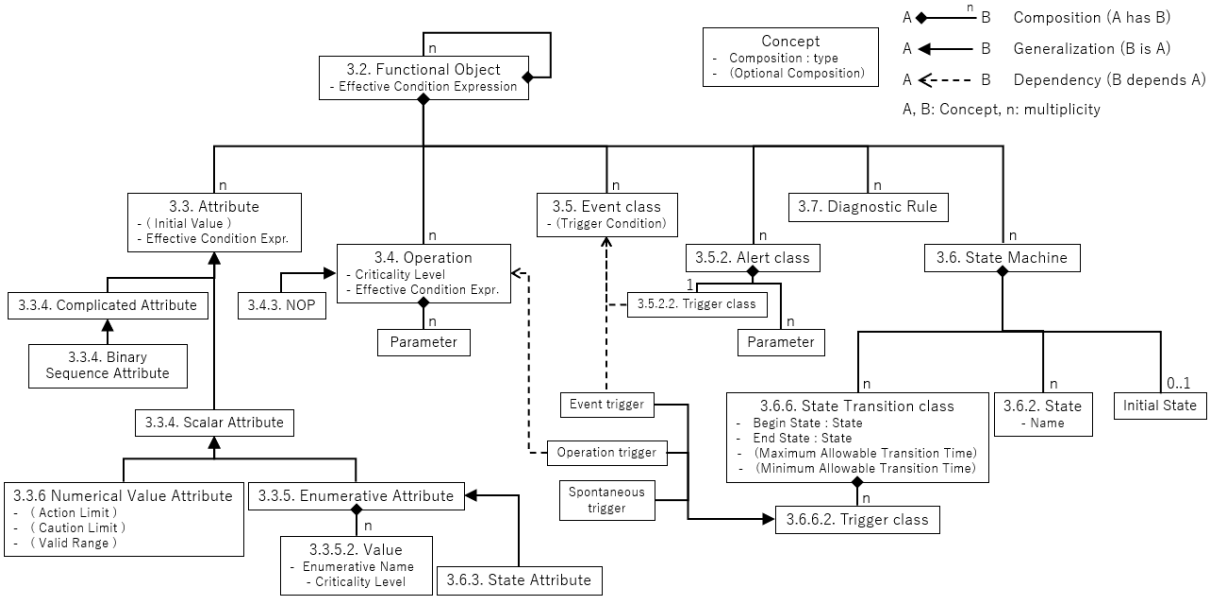

Figure 3-1 Summary of the Concepts to Specify Functional Objects
Functional Objects を定めるのに用いる概念のサマリ

To specify various conditions including the Effective-Condition of a Functional Object, Attribute or Operation (see Sections 3.2.3, 3.3.2, and 3.4.5) or the Trigger Condition of an Event class, an expression referred to as a Condition Expression is used (see Section 3.9).

Functional Object, Attribute, や Operation の有効条件（3.2.3 項、3.3.2 項、及び 3.4.5 項参照）や Event class の Trigger Condition（3.5.1.2 項参照）等、各種の条件を定めるのに、条件式と称する式を用いる（3.9 項参照）。

A signal for communication to a Functional Object from another entity is referred to as a telecommand message and that from a Functional Object to another entity is referred to as a telemetry message in this model. An entity which sends telecommand messages is referred to as a Controller.

本モデルでは、ある Functional Object へ他の構成要素からの通信に用いる信号を telecommand message, ある Functional Object から他の構成要素への通信に用いる信号を telemetry message と称する。また、telecommand messages を送信する構成要素を Controller と称する。

3.2. FUNCTIONAL OBJECTS

3.2.1. General // 一般

An entire spacecraft **shall** be represented by one Functional Object which contains one or more Functional Objects. Any Functional Object of a spacecraft is either directly or indirectly contained in the Functional Object that represents the entire spacecraft.

宇宙機全体は一つ以上の Functional Objects を持つ一つの Functional Object で表されること。ある宇宙機のいかなる Functional Object も、その宇宙機全体を表す Functional Object に、直接的か間接的かの何れかで含まれる。

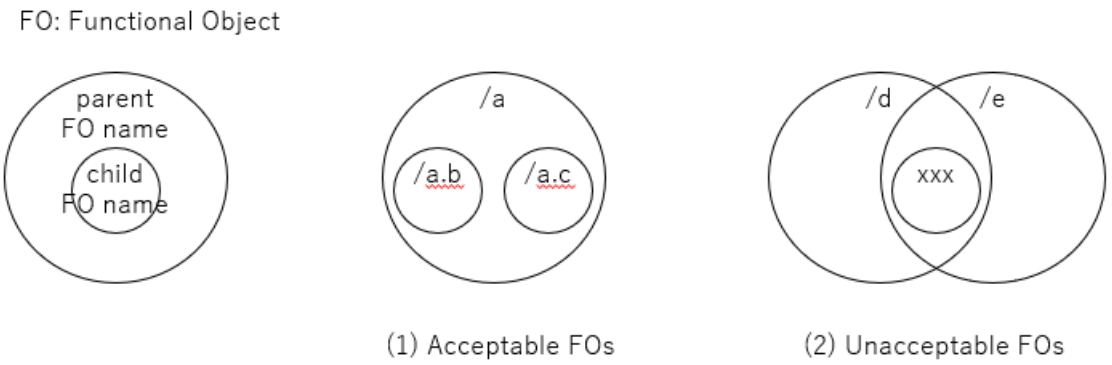

Figure 3-2 Acceptable / Unacceptable Combination of Parent-Child Relation between Functional Objects // Functional Objects の親子関係で許容される・許容されない組み合わせ

The names of the Functional Objects follow the convention in [R1]. For xxx, a name cannot be allocated according to the convention.

Functional Objects の名前は、[R1] の規則に従う。xxx には、規則に従い名前を割り当てる事はできない。

3.2.2. Parent-Child Relation // 親子関係

A Functional Object **shall** contain zero or more Functional Objects.

Functional Object は、Functional Objects をゼロ個以上含む**こと**。

[Example 1] Functional Object /a in Figure 3-2 (1) contains Functional Object /a.b and Functional Object /a.c.

[例 1] Figure 3-2 (1) の Functional Object /a は、Functional Object /a.b と Functional Object /a.c を含む。

A Functional Object **shall** not be directly contained in two or more Functional Objects.

Functional Object は、二つ以上の Functional Objects には直接的に含まれない**こと**。

[Example 2] Functional Object d and Functional Object e in Figure 3-2 (2) contain Functional Object f, which is not allowed.

[例 2] Figure 3-2 (2) の Functional Object d と Functional Object e は、Functional Object f を含むが、これは許されない。

When a Functional Object contains one or more Functional Objects in it, the former is referred to as the "parent Functional Object" of the latter and the latter are referred to as "child Functional Objects" of the former. Furthermore, generalized terms of these are introduced; the Functional Objects at any upper and lower layers of the hierarchy of the parent/child relations, such as a parent of a parent (upper layer) and a child of a child (lower layer), are referred to as "ancestor Functional Object(s)" and "descendant Functional Object(s)", respectively.

ある Functional Object に一つ以上の Functional Objects が含まれる場合、前者は後者の「親 Functional Object」と称し、後者は前者の「子 Functional Objects」と称する。さらに、これらを一般化した用語を導入する。 親の親（上位階層）、子の子（下位階層）等、親/子関係の任意の上位階層及び下位階層にある Functional Objects は、それぞれ、「祖先 Functional Object(s)」及び「子孫 Functional Object(s)」と称する。

3.2.3. Effective-Conditions of Functional Objects (Valid/Invalid and Effective-Condition Expressions) Functional Objects の有効条件（有効/無効及び有効条件式）

A Functional Object (except for the Functional Object representing an entire spacecraft) **shall** be either valid or invalid at a given time. When a Functional Object is invalid, it **shall** suspend all of its functions.

Functional Object（宇宙機全体を表す Functional Object を除く）は、ある時点に有効か無効の何れかであること。ある Functional Object が無効な場合、その機能の全てを停止していること。

The condition which determines whether a Functional Object is valid or invalid at a given time is referred to as the Effective-Condition of the Functional Object.

ある時点に Functional Object が有効か無効かを決める条件を、その Functional Object の有効条件と称する。

For each Functional Object except for the Functional Object representing the entire spacecraft, its Effective-Condition **shall** be specified with a Condition Expression referred to as the Effective-Condition Expression.

宇宙機全体を表す Functional Object を除く各 Functional Object に、有効条件式と称する条件式にて、その有効条件を定めること。

The Functional Object representing an entire spacecraft **shall** be always valid. As for the other Functional Objects, one **shall** be valid when the evaluation result of its Effective-Condition is true and when its parent Functional Object is valid, or else it **shall** be invalid.

宇宙機全体を表す Functional Object は、常に有効であること。その他の Functional Object は、その有効条件式を評価した結果が真であり、かつ、親 Functional Object が有効である場合、有効であること。さもなくければ無効であること。

3.3. ATTRIBUTES

3.3.1. General // 一般

A variable that represents a status of a Functional Object at a given time is referred to as an Attribute of the Functional Object. The values of Attributes of a Functional Object at a given time **should** be able to be obtained by other entities using a telemetry message. A Functional Object **shall** have zero or more Attributes.

ある Functional Object のある時点における状況を表す変数を、その Functional Object の Attribute と称する。ある時点における Functional Object の Attributes の値は、telemetry message を用いて、他の構成要素によって取得できる**べき**である。Functional Object は、ゼロ個以上の Attributes を持つ**こと**。

[Note 1] In [R3], the values of Attributes are contained in a telemetry message called a VALUE Telemetry. The values of Attributes are also contained in a telemetry message called a NOTIFICATION Telemetry.

[注 1] [R3] では、Attributes の値は、VALUE Telemetry と呼ばれる telemetry message に含める。また、Attributes の値は、NOTIFICATION Telemetry と呼ばれる telemetry message にも含める。

[Note 2] In [R3], the values to be set for Attributes are contained in a telemetry message called a SET Telecommand or an ACTION Telecommand.

[注 2] [R3] では、Attributes に設定する値は SET Telecommand または ACTION Telecommand と呼ばれる telecommand message に含める。

[Note 3] In [R3], a VALUE Telemetry is generated by a Functional Object spontaneously or in response to a request in a telecommand message called GET Telecommand sent from a Controller.

[注 3] [R3] では、VALUE Telemetry は、Functional Object によって、自発的に、または、Controller から送信された GET Telecommand と呼ばれる telecommand message の要求に応じて、生成される。

The values of some Attributes of a Functional Object can be set by Controllers. In order for a value of an Attribute of a Functional Object to be set by a Controller, an Operation of the Functional Object **shall** be invoked (see Sections 3.4.2 and 3.4.4.1). Whether the value of an Attribute has been set correctly or not by a Controller can be verified by the Controller by inspecting the telemetry value of the Attribute.

Functional Object の Attributes の一部は、Controllers が値を設定できる。ある Functional Object の Attribute の値を Controller が設定するには、その Functional Object の Operation を呼び出す**こと**(3.4.2 項及び 3.4.4.1 項参照)。ある Controller がある Attribute の値を正常に設定したか否かは、その Controller がその Attribute のテレメトリ値を調べる事で検証できる。

3.3.2. Effective-Conditions of Attributes (Valid/Invalid and Effective-Condition Expressions)
Attributes の有効条件（有効/無効及び有効条件式）

An Attribute **shall** be either valid or invalid at a given time.

Attribute は、ある時点に有効か無効の何れかであること。

When an Attribute is valid, an Attribute **shall** take one of the values in the set specified for the Attribute.

ある Attribute が有効である場合、ある Attribute は、その Attribute に定められた集合内の値の何れかをとること。

When an Attribute is invalid, its value has no significance and cannot be reset by a Controller. In that case, the Functional Object that has the Attribute **shall** ignore requests received from Controllers.

ある Attribute が無効である場合、その値は意味を持たず、Controller から再設定する事はできない。その場合、その Attribute を持つ Functional Object は Controllers から受信した要求を無視すること。

A condition which determines whether an Attribute is valid or invalid at a given time is referred to as an Effective-Condition of the Attribute.

ある時点に Attribute が有効か無効かを決める条件を、その Attribute の有効条件と称する。

For each Attribute, an Effective-Condition **shall** be specified with a Condition Expression referred to as an Effective-Condition Expression.

各 Attribute に、有効条件式と称する条件式にて有効条件を定めること。

An Attribute of a Functional Object at a given time **shall** be valid when the Functional Object is valid and when the evaluation result of the Effective-Condition Expression of the Attribute is true, or else it **shall** be invalid.

ある Functional Object のある Attribute は、ある時点で、その Functional Object が有効であり、かつ、その Attribute の有効条件式を評価した結果が真である場合、有効であること。さもなくければ無効であること。

3.3.3. Criticality Levels of values

For the range or set of the values that each Attribute can take, a Criticality Level **shall** be specified. The Criticality Levels are used by an entity that monitors and controls a Functional Object to diagnose whether the Functional Object is functioning correctly or not.

それぞれの Attribute が取り得る値の範囲または集合に対して、Criticality Level を定めること。Criticality Level は、ある Functional Object を監視制御する構成要素が、その Functional Object が正常に機能しているか否かの診断に用いる。

The Criticality Level of a value **shall** be one of Action, Caution, and Normal.
- When the Criticality Level of the value of an Attribute of a Functional Object is Action, it **shall** indicate that the Functional Object is in danger (for example, the Functional Object approaches permanent failure).
- When the Criticality Level of the value of an Attribute of a Functional Object is Caution, it **shall** indicate that caution is needed for the Functional Object (for example, the Functional Object has temporarily ceased functioning).
- When the Criticality Level of the value of an Attribute of a Functional Object is neither Action nor Caution, the Criticality Level is Normal.

ある値の Criticality Level は、Action, Caution, 及び Normal の何れかであること。
- ある Functional Object の Attribute の値の Criticality Level が、Action である場合、その Functional Object が危険である事（例えばその Functional Object が恒久故障に向かっている事）を示すこと。
- ある Functional Object の Attribute の値の Criticality Level が、Caution である場合、その Functional Object には注意が必要である事（例えば Functional Object が一時的に機能停止している事）を示すこと。
- ある Functional Object の Attribute の値の Criticality Level が、Action でも Caution あるでもない場合、その Criticality Level は、Normal である。

3.3.4. Initial Value

An Attribute **should** have a specific value referred to as an Initial Value. If an Attribute has the Initial Value, the Attribute **shall** take the value of the Initial Value when it becomes valid. If an Attribute does not have an Initial Value, which value in the specified set the Attribute takes is not predictable when it becomes valid.

Attribute は Initial Value と称する特定の値を持つ**べき**である。ある Attribute が Initial Value を持つ場合、その Attribute は、有効になった際に Initial Value の値を取る**こと**。ある Attribute が Initial Value を持たない場合、有効になったときに、定められた集合内の値の何れをとるかは予測不可能である。

3.3.5. Scalar Attributes and Complicated Attributes

An Attribute **shall** take one of the values in the set specified for the Attribute. An Attribute which takes a value of a simple data type (such as an enumeration type or integer) is referred to as a Scalar Attribute. An Attribute which takes a value of a more complicated data type (such as an array or structure type) is referred to as a Complicated Attribute.

ある Attribute は、その Attribute に定められた集合内の値の何れかをとる**こと**。単純なデータ型（列挙型や整数等）の値をとる Attribute を Scalar Attribute と称する。また、より複雑なデータ型（配列型や構造体型等）の値をとる Attribute を Complicated Attribute と称する。

A Complicated Attribute whose contents is not modeled in the FMS is referred to as Binary Sequence Attribute.

Complicated Attribute のうち、FMS ではその内容をモデル化しないものを Binary Sequence Attribute と称する。

[Note] Except for Binary Sequence Attribute, complicated data types have not been implemented in the tools of SIB2/GSTOS yet. Their proxy is under review.

[注] Binary Sequence Attribute を除き、複雑なデータ型は SIB2/GSTOS のツールでまだ実装されていないが、代替策を検討中。

3.3.6. Enumerative Attribute

3.3.6.1. General // 一般

A Scalar Attribute whose value (integer number) is labeled with a name (identifier consisting of alphabets and numbers) is referred to as an Enumerative Attribute.

値（整数値）が名前（アルファベット及び数字で構成される識別子）でラベルされた Scalar Attribute を Enumerative Attribute と称する。

3.3.6.2. Values and Enumerative Names // 値及び Enumerative Names

Enumerative Attributes **shall** have discrete values (such as the current mode of an instrument). For an Enumerative Attribute, the Valid Value Set, which is the set of the values that the Enumerative Attribute can take, **shall** be specified. For each of the values, a name referred to as an Enumerative Name **shall** be specified.

Enumerative Attributes は、離散値（機器の現在のモード等）を持つ**こと**。ある Enumerative Attribute に対して、その Enumerative Attribute が取り得る値の集合である Valid Value Set を定める**こと**。また、これらの値の各々に Enumerative Name と称する名前を定める**こと**。

When an Enumerative Attribute is valid, the value of the Enumerative Attribute **shall** be one of the values in the Valid Value Set of the Enumerative Attribute. When an Enumerative Attribute is invalid, the value of the Enumerative Attribute **may** not be one of the values in the Valid Value Set of the Enumerative Attribute.

Enumerative Attribute の値は、その Enumerative Attribute が有効な場合、その Enumerative Attribute の Valid Value Set の内の値の一つである**こと**。Enumerative Attribute の値は、その Enumerative Attribute が無効な場合、その Enumerative Attribute の Valid Value Set の内の値の一つでなくて**良い**。

3.3.6.3. Criticality Level

For a value in the Valid Value Set of an Enumerative Attribute, a Criticality Level shall be specified (See Section 3.3.3). The Criticality Level of a value is one of Action, Caution, and Normal.

Enumerative Attribute の Valid Value Set 内の値に対して、Criticality Level を定める**こと** (3.3.3 項参照)。ある値の Criticality Level は、Action, Caution, 及び Normal の何れかである。

[Note] Diagnosis with more complexed conditions can be specified with Diagnostic Rules (See Section 3.7).

[注] より複雑な条件による診断を、Diagnostic Rules にて定める事ができる（3.7 項参照）。

3.3.7. Numerical Value Attribute

3.3.7.1. General // 一般

A Scalar Attribute whose values are not labeled with names is referred to as a Numerical Value Attribute.

値が名前でラベルされていない Scalar Attribute を Numerical Value Attribute と称する。

A Numerical Value Attribute **shall** have either a continuous value (*e.g.* a temperature) or a discrete value (*e.g.* values of a counter: 0,1,2, …).

Numerical Value Attribute は、連続値（例えば、温度）か離散値（例えば、カウンタ値：0,1,2, …）の何れか一方を持つ**こと**。

3.3.7.2. Valid Ranges

A pair of upper and lower limits, referred to as Valid Range, **may** be specified for a Numerical Value Attribute. If a Valid Range is specified for a Numerical Value Attribute, the Numerical Value Attribute **shall** take a value within its Valid Range (for example, due to physical limitation of the sensor that measures the value of the Attribute).

Numerical Value Attribute には Valid Range と称する一組の上限値と下限値を定めて**良い**。ある Numerical Value Attribute に Valid Range を定めた場合、その Numerical Value Attribute は、その Valid Range 内の値のみを取る**こと**（たとえば、その Attribute の値を測定する装置の物理的制約により）。

When a Numerical Value Attribute is valid, the value of the Numerical Value Attribute **shall** be in the Valid Rang of the Numerical Value Attribute. When a Numerical Value Attribute is invalid, the value of the Numerical Value Attribute **may** not be in the Valid Rang of the Numerical Value Attribute.

Numerical Value Attribute の値は、その Numerical Value Attribute が**有効**な場合、その Numerical Value Attribute の Valid Range 内である**こと**。Numerical Value Attribute の値は、その Numerical Value Attribute が**無効**な場合、その Numerical Value Attribute の Valid Range 内でなくて**良い**。

3.3.7.3. Action Limits, Caution Limits, and Criticality Level

Another pair of upper and lower limits, referred to as an Action Limit, **may** be specified. For a Numerical Value Attribute, another pair of upper and lower limits, referred to as a Caution Limit, **may** also be specified.

Numerical Value Attribute には、Action Limit と称する別の一組の上限値と下限値を定めて**良い**。Numerical Value Attribute には、Caution Limit と称する別の一組の上限値と下限値も定めて**良い**。

These limit values are used by an entity that monitors and controls a Functional Object to diagnose whether the Functional Object is functioning correctly or not.

これらの限界値は、ある Functional Object を監視制御する構成要素が、その Functional Object が正常に機能しているか否かの診断に用いる。

When the value of a Numerical Value Attribute falls out of the range of its Action Limit, the Criticality Level of the value **shall** be defined as Action. When the value of the Numerical Value Attribute falls within its Action Limit, but out of the range of its Caution Limit, the Criticality Level of the value **shall** be defined as Caution. Otherwise, the Criticality Level of the value **shall** be defined as Normal.

ある Numerical Value Attribute の値が、その Action Limit の範囲を外れた場合、その値の Criticality Level は Action と定義される**こと**。その Numerical Value Attribute の値が、その Action Limit の範囲に収まるも Caution Limit の範囲を外れる場合、その値の Criticality Level は Caution と定義される**こと**。さもなければ、その値の Criticality Level は Normal と定義される**こと**。

[Note] Diagnosis with more complexed conditions can be specified with Diagnostic Rules (See Section 3.7).

[注] より複雑な条件による診断を、Diagnostic Rules にて定める事ができる（3.7 項参照）。

3.4. OPERATIONS

3.4.1. General // 一般

An action performed by a Functional Object (*e.g.* to set a value to an Attribute) is referred to as an Operation of the Functional Object. A Functional Object **shall** have zero or more Operations.

ある Functional Object が実行する動作（例えば、Attribute の値を設定する）を、その Functional Object の Operation と称する。Functional Object は、ゼロ個以上の Operations を持つこと。

[Note 1] Most Functional Objects have one or more Operations.

[注 1] ほとんどの Functional Objects は、1 つ以上の Operations を持つ。

An Operation of a Functional Object **shall** be invoked when the Functional Object receives a telecommand message from a Controller.

Functional Object の Operation は、その Functional Object が Controller から telecommand message を受信した時に呼び出されること。

[Note 2] In [R3], an Operation is invoked with a telecommand message called an ACTION Telecommand. An Operation is also invoked with a telecommand message called a SET Telecommand.

[注 2] [R3] では、Operation を、ACTION Telecommand と呼ばれる telecommand message で呼び出す。また、Operation は、SET Telecommand と呼ばれる telecommand message でも呼び出す。

3.4.2. Attribute Change Rules

The rule for changes of Attribute values after an execution of an Operation is referred to as the Attribute Change Rule of the Operation.

ある Operation の実行後の Attributes の変化の規則を、その Operation の Attribute Change Rule と称する。

Functional Objects **should** be designed in a way that the values of one or more Attributes change as a result of execution of an Operation except for the cases in which it is physically impossible to do so due to design constraints. If so designed, whether an Operation has been executed correctly or not can be verified by checking whether the Attribute values that are supposed to change as a result of the execution of the Operation have actually changed or not.

Functional Objects は、設計上の制約のために物理的に不可能な場合を除いて、ある Operation の実行結果として、一つ以上の Attributes の値が変化するように設計すべきである。そのように設計された場合、ある Operation が正常に実行されたか否かは、その Operation の実行の結果変わるはずの Attributes 値が、実際に変わったかどうかを確認する事で検証できる。

For an Operation, zero or more Attributes whose values are predictable constant values after execution of the Operation **shall** be specified. For each of the Attributes, a constant value after execution of the Operation **shall** be specified. The entity which sends a telecommand message to invoke an Operation **should** verify that the values of the Attributes are the specified constant values after the execution of the Operation, referring to telemetry messages.

Operation に対して、その Operation の実行後に値が予測可能な定数値となる Attributes をゼロ個以上定めること。 その Attributes の各々に対して、Operation の実行後の定数値を定めること。ある Operation を呼び出す telecommand message を送る構成要素は、その Operation の実行後、telemetry messages を参照し、Attributes の値が指定された定数値である事を確認すべきである。

[Note] For some Operations of a Functional Object, Current States of its State Machines (see Section 3.6.2) also change as a result of their execution. The Current States of a Functional Object are indicated by the values of the State Attributes of the Functional Object (see Section 3.6.2).

[注] ある Functional Object の一部の Operations では、その実行の結果としてその State Machines の Current States (3.6.2 項参照) も変わる。ある Functional Object の Current States は、その Functional Object の State Attributes の値で示される (3.6.2 項参照)。

3.4.3. No Operation (NOP)

An Operation which accompanies no changes is referred to as No Operation (NOP).

変化を伴わない Operation を No Operation (NOP) と称する。

Each Functional Object **should** have a NOP to check the health of the communication channel.

通信路の健全性の確認のため、各 Functional Object は NOP を持つべきである。

3.4.4. Value Setting Attributes and Parameters of Operations

3.4.4.1. General // 一般

For an Operation of a Functional Object, zero or more Attributes and zero or more parameters referred to as Value Setting Attributes or Parameters, respectively, **shall** be specified. A telecommand message to invoke the Operation **shall** contain the values to be set to the Value Setting Attributes specified for the Operation and the values of the Parameters specified for the Operation. The values of the Parameters **shall** describe detailed information on how the Operation is executed. The Functional Object **shall** set the received values to the Value Setting Attributes when the Operation is invoked. The entity which sends a telecommand message to invoke the Operation **should** verify that the values of the Value Setting Attributes are the sent values after the execution of the Operation, referring to telemetry messages.

Functional Object の Operation に、ゼロ個以上の Value Setting Attributes と称する Attributes 及びゼロ個以上の Parameters と称するパラメータを定めること。ある Operation を呼び出す telecommand message は、その Operation に定めた Value Setting Attributes に設定する値、並びに、その Operation に定めた Parameters の値を含むこと。これらの Parameters の値は、この Operation を如何に実行するかの詳細情報を記すこと。また、その Functional Object は、この Operation が呼び出されたときに、これらの Value Setting Attributes に受け取った値を設定すること。その Operation を呼び出す telecommand message を送る構成要素は、その Operation の実行後、telemetry messages を参照し、Value Setting Attributes の値が送った値である事を確認すべきである。

3.4.4.2. Valid Ranges

A pair of upper and lower limits, referred to as a Valid Range, **may** be specified for a Parameter. If a Valid Range is specified for a Parameter, Controllers **should** only send a telecommand message which contains a value of the Parameter within the Valid Range. Similarly, if a Valid Range is specified for an Value Setting Attribute (see Section 3.3.7.2), Controllers **should** only send a telecommand message which contains a value of the Value Setting Attribute within the Valid Range.

Parameter には Valid Range と称する別の一組の上限値と下限値を定めて良い。ある Parameter に Valid Range を定めた場合、Controllers は、その Parameter が Valid Range 内の値を含む telecommand message のみを送るべきである。同様に、ある Value Setting Attribute に Valid Range を定めた場合 (3.3.7.2 項参照)、Controllers は、その Value Setting Attribute の Valid Range 内の値を含む telecommand message のみを送るべきである。

3.4.5. Effective-Conditions of Operations (Valid/Invalid and Effective-Condition Expressions) Operations の有効条件（有効/無効及び有効条件式）

An Operation **shall** be either valid or invalid at a given time. When an Operation is valid, the Operation **shall** be executed. When an Operation is invalid, Controllers (including onboard Controllers, desirably) **should** not send a telecommand message to execute the Operation. Controllers on the ground **shall** have a function to warn of sending of a telecommand message to execute an Operation when the Operation is invalid. If the Functional Object which has an Operation has received a telecommand message to execute the Operation, the Functional Object **shall** ignore the telecommand message.

Operation は、ある時点に有効か無効の何れかであること。ある Operation が有効である場合、その Operation は実行されること。ある Operation が無効である場合、Controllers（できれば宇宙機搭載の Controllers も含めて）はその Operation を実行するための telecommand message を送るべきではない。地上の Controllers は、ある Operation が無効である場合、その Operation を実行するための telecommand message の送信に対して、警告をする機能をもつこと。ある Operation を持つ Functional Object は、その Operation を実行するための telecommand message を受信した場合、その telecommand message を無視すること。

The condition which determines whether an Operation is valid or invalid at a given time is referred to as an Effective-Condition of the Operation.

ある時点に Operation が有効か無効かを決める条件を、その Operation の有効条件と称する。

An Effective-Condition **shall** consist of a Condition Expression and a condition of Effective States (see Section 3.6.6). The Condition Expression is referred to as the Effective-Condition Expression.

有効条件は、Effective States（3.6.6 項参照）の条件と条件式からなること。この条件式を有効条件式と称する。

For each Operation, an Effective-Condition Expression **shall** be specified.

各 Operation に、有効条件式を定めること。

An Operation of a Functional Object at a given time **shall** be valid when the Functional Object is valid, when the evaluation result of the Effective-Condition Expression of the Operation is true, and when all the State Machines which have Effective States for the Operation are in one of the States classified as the Effective States for the Operation, or else it **shall** be invalid.

ある Functional Object のある Operation は、ある時点で、その Functional Object が有効であり、かつ、その Operation の有効条件式を評価した結果が真であり、かつ、その Operation に対する Effective States を持つ全ての State Machines がその Operation の Effective States に分類される State の何れかにある場合、有効であること。さもなくければ無効であること。

3.4.6. Criticality Level

For an Operation, its Criticality Level **shall** be specified. A Criticality Level **shall** be either prohibited, warning, or normal.

Operation に、Criticality Level を定めること。Criticality Level は、prohibited, warning, または normal の何れかであること。

If the Criticality Level of an Operation is prohibited, the Operation **shall** not be executed. Controllers on the ground **shall** have a function to prevent sending telecommand messages to execute Operations whose Criticality Levels are prohibited. Controllers aboard a spacecraft **should** have a function to prevent sending telecommand messages to execute Operations whose Criticality Levels are prohibited, desirably.

ある Operation の Criticality Level が prohibited である場合、その Operation は実行されないこと。地上の Controllers は、Criticality Levels が prohibited である Operation を実行するための telecommand messages の送信を防ぐ機能を持つこと。宇宙機搭載の Controllers は、できれば、Criticality Levels が prohibited である Operation を実行するための telecommand messages の送信を防ぐ機能を持つべきである。

If the Criticality Level of an Operation is warning, caution **shall** be need for execution of the Operation. Controllers on the ground **shall** have a function to call for attention when sending telecommand messages to execute Operations whose Criticality Levels are warning.

ある Operation の Criticality Level が warning である場合、その Operation を実行には注意をすること。地上の Controllers は、Criticality Levels が warning である Operation を実行するための telecommand messages の送信の際に注意を喚起する機能を持つこと。

3.5. EVENT CLASSES

3.5.1. Event and Event classes

3.5.1.1. General // 一般

A classification of an event (an occurrence of a thing that has a particular significance) that occurs in a Functional Object is referred to as an Event class of the Functional Object.

ある Functional Object で発生する event（特定の意味を有する出来事の発生） の分類を、その Functional Object の Event class と称する。

A Functional Object **shall** have zero or more Event classes.

Functional Object は、ゼロ個以上の Event classes を持つこと。

An event **shall** belong to one Event class.

ある event は、一つの Event class に属すること。

Eevents of some Event classes are detected by a Functional Object (see Section 3.5.2.1).

一部 Event classes の events は、Functional Object で検出する (3.5.2.1 項参照)。

Event classes are used to specify Trigger classes of State Transition classes (see Section 3.6.6) and/or the Trigger classes of Alert classes (see Section 3.5.2.1).

Event classes は、State Transition classes の Trigger classes (3.6.6 項参照)、Alert classes の Trigger classes (3.5.2.1 項参照) の一方または双方の指定に用いる。

3.5.1.2. Trigger Conditions

For occurrence of an event of an Event class, an explicit condition referred to as a Trigger Condition **may** be specified.

- If a Trigger Condition is specified for an event of an Event class, it **shall** be specified by a Condition Expression. An event of the Event class **shall** be triggered when the evaluation result of its Condition Expression becomes true.
- If an explicit condition is not specified for trigger of an event of an Event class, an event of the Event class is triggered by some unidentified internal activity of the Functional Object.

ある Event class の event の発生に、Trigger Condition と称する明確な条件を定めて良い。

- ある Event class の event に Trigger Condition を定める場合は、条件式で指定すること。その Event class の event は、その条件式を評価した結果が真になったときにトリガされること。
- ある Event class の event のトリガに明確な条件を定ない場合は、その Event class の event は、Functional Object 内部の何らかの未知の活動によって、トリガされる。

3.5.2. Alerts and Alert classes

3.5.2.1. General // 一般

Functional Objects can report to other entities an event of an Event class that is important to them. The report is referred to as an alert and its classification is referred to as an Alert class.

Functional Objects は、他の構成要素にとり重要な Event class の event をそれらに通知できる。この通知を alert と称し、その分類を Alert class と称する。

If a Functional Object has a function to notify an event as an alert, an Event class to which the notified event belongs and an Alert class to which the corresponding alert belongs **shall** be specified.

ある Functional Object に event を alert として通知する機能を持たせる場合、通知する event が属する Event class と対応する alert が属する Alert class を定めること。

A Functional Object **shall** have zero or more Alert classes.

Functional Object は、ゼロ個以上の Alert classes を持つこと。

An alert **shall** belong to an Alert class.

ある alert は、一つの Alert class に属すること。

An alert of a Functional Object is transferred to other entities with a telemetry message.

ある Functional Object の alert は、telemetry message で他の構成要素に伝送される。

[Note 1] A telemetry message which transfers an alert is called a NOTIFICATION Telemetry in [R3].

[注 1] Alert を伝送する telemetry message を、[R3] では、NOTIFICATION Telemetry と呼ぶ。

[Note 2] In some cases, an entity can detect an event of an Event class by monitoring the values of some relevant Attributes periodically. However, an event is not necessary detected by the entity because the values of the Attributes are not always delivered to the entity in a sufficiently frequent manner. By contrast, the alert reports an event actively and more promptly.

[注 2] ある構成要素は、幾つかの関連する Attributes 値を定期的に監視する事である Event class の event を検出できる場合がある。しかし、これらの Attributes の値は、その構成要素に必ずしも十分頻繁に配信されないため、必ずしもその構成要素は event を検出できるとは限らない。対照的に、alert は event を能動的かつより迅速に通知する。

3.5.2.2. Triger classes of Alerts classes

The classification of the events which are the triggers of the reports of the alerts of an Alerts class is referred to as the Triger class of the Alerts class. One Event class **shall** be specified as the Triger class for an Alert class.

ある Alerts class の alerts の通知のトリガとなる events の分類を、その Alerts class の Triger class と称する。ある Alert class に対して、Triger class として一つの Event class を定めること。

The Functional Object **shall** detect an event of the Trigger class and report to other entities with an alert of the Alert class. The Functional Object which detects events of a Trigger class may or may not be the Functional Object for which the Trigger class is specified.

その Functional Object は、その Trigger class の event を検出し、その Alert class の alert にて、他の構成要素に通知すること。ある Trigger classes の events を検出する Functional Object は、その Trigger classes を定めた Functional Object であっても良いし、なくても良い。

3.5.2.3. Parameters and Value Notifying Attributes of Alerts classes

For an Alert class of a Functional Object, zero or more Attributes and zero or more parameters referred to as Value Notifying Attributes and Parameters, respectively, **shall** be specified. An alert classified as an Alert class **shall** contain the values of the Value Notifying Attributes and Parameters specified for the Alert class. These values of the Value Notifying Attributes **shall** be the values at the timing of the event and the values of Parameters **shall** describe the detail of the event.

Functional Object の Alert class に、ゼロ個以上の Value Notifying Attributes と称する Attributes 及びゼロ個以上の Parameters と称するパラメータを定めること。ある Alert class に分類される alert は、その Alert class に定めた Value Notifying Attributes 及び Parameters の値を含むこと。これらの Value Notifying Attributes の値は event の時点での値であり、また、Parameters の値は、event の詳細を記すこと。

3.6. STATE MACHINES

3.6.1. General // 一般

A Functional Object **shall** include zero or more State Machines.

Functional Object は、ゼロ個以上の State Machines を含むこと。

3.6.2. States, Names of States, and Current State

A state which a State Machine has is referred to as a State. A State Machine **shall** have two or more States.

State Machine が持つ状態を State と称する。State Machine は、二つ以上の States を持つこと。

A State **shall** have one Name (name).

State は、一つの Name（名前）を持つこと。

The State which a State Machine currently takes is referred to as a Current State.

ある State Machine が現在取る State を Current State と称する。

3.6.3. State Attributes

A Functional Object **shall** have one Enumerative Attribute corresponding to each of its State Machines, referred to as a State Attribute.

Functional Object は、その State Machines のそれぞれに、State Attribute と称する Enumerative Attribute を一つ持つこと。

In Figure 3-3, the concepts associated with State Attributes and State Machines are summarized. The concepts associated with State Attributes have 1-to-1 relation with the concepts associated with State Machines. The concepts of State Attributes and the concepts of State Machines represent the same thing from different point of views.

Figure 3-3 に State Attributes と State Machines に関する概念を要約する。State Attributes に関する概念は、State Machines に関する概念と一対一の関係にある。State Attributes に関する概念と State Machines に関する概念は同じものを異なる観点から表現したものである。

The value which the State Attribute of a State Machine takes **shall** indicate the State of the State Machine.

ある State Machine に対する State Attribute が取る値は、その State Machine の State を示すこと。

The Enumerative Name of a value of the State Attribute corresponding to a State Machine **shall** be the Name of a State which corresponds to the value.

ある State Machine に対応する State Attribute の値の Enumerative Name は、その値に対応する State の Name であること。

[Note 1] The Current States of the State Machines of a Functional Object determine a set of the Operations that can be invoked at the time (see Section 3.6.6.4).

[注 1] ある Functional Object の State Machines の Current States は、その時点で呼び出す事ができる Operations の集合を決める（3.6.6.4 項参照）。

[Note 2] A Criticality Level is specified for a value of a State Attribute (see Section 3.3.6.3).

[注 2] State Attribute の値には Criticality Level を定める（3.3.6.2 項参照）。

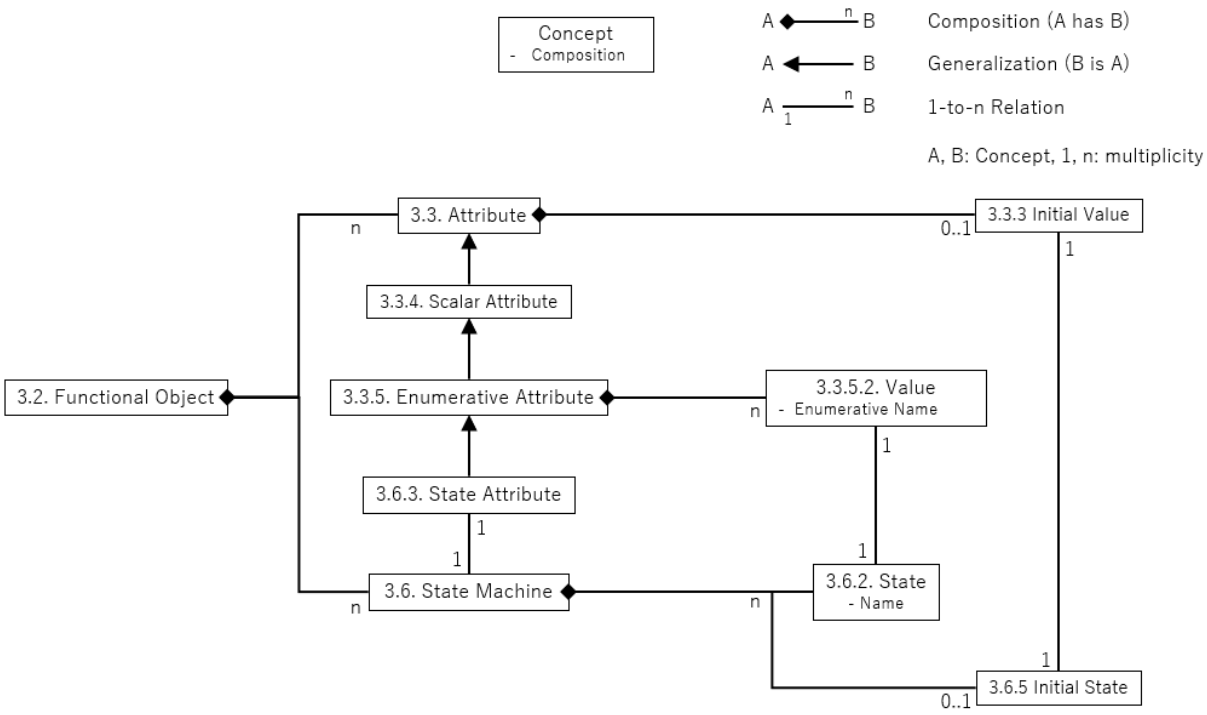

Figure 3-3 The Concepts Associated with State Attributes and State Machines
State Attributes と State Machines に関する概念

The concepts associated with State Attributes and State Machines are extracted from Figure 3-1 and the relations between the concepts associated with State Attributes and those associated with State Machines are added. In this figure, compositions and 1-to-1 relations are shown in the vertical direction and generalizations are shown in the horizontal direction.

Figure 3-1 から、State Attributes と State Machines に関する概念を抽出し、State Attributes に関する概念と State Machines に関する概念の関係を追加した。本図では、コンポジションと 1 対 1 の関連を垂直方向に示し、汎化を水平方向に示す。

3.6.4. Valid/Invalid // 有効/無効

A State Machine **shall** be either valid or invalid at a given time. Whether a State Machine is valid or not **shall** be identical with that of the State Attribute (see Section 3.6.2) corresponding to the State Machine, which means some State Machines of a Functional Object are valid whenever the Functional Object is valid.

State Machine は、ある時点に有効か無効の何れかであること。ある State Machine が有効か否かは、その State Machines に対応する State Attribute (3.6.2 項参照) が有効か否かと同一であること。この事は、ある Functional Object の一部の State Machines は、その Functional Object が有効な場合は常に有効である事を意味する。

3.6.5. Initial State

If a State Machine is always in a specific State when the State Machine becomes valid, the State is referred to as the Initial State. Depending on whether the State Attribute for a State Machine has an Initial Value or not, the State Machine has or does not have an Initial State, respectively. If a State Machine does not have an Initial State, in which State the State Machine is when the State Machine becomes valid is not predictable.

ある State Machine が有効になったときに、その State Machine が常に特定の State になる場合、その State を Initial State と称する。ある State Machine に対応する State Attribute が Initial Value をもつか否かに応じ、それぞれ、その State Machine は、Initial State を持つか持たない。ある State Machine に Initial State を持たない場合、その State Machine が有効になったときに、その State Machine の State の何れになるかは予測不可能である。

3.6.6. Transitions and State Transition classes // 遷移及び State Transition classes

3.6.6.1. General // 一般

A pattern of transition of the State of a State Machine from one State to another State is referred to as a State Transition class.

State Machine の State が、ある State から他の State に遷移するパターンを State Transition class と称する。

A State Machine **shall** have one or more State Transition classes.

State Machine は、一つ以上の State Transition classes を持つこと。

Transition from one State to another State **shall** belong to a State Transition class.

ある State から他の State に遷移は、一つの State Transition class に属すること。

A State Transition class **shall** have a Begin State (source State in a transition) and an End State (target State in a transition).

State Transition class は、Begin State（遷移元の State）と End State（遷移先の State）を持つこと。

A transition of a State Transition class occurs when it is triggered by 1) execution of an Operation, 2) an event, or 3) some internal activity of the Functional Object.

State Transition class の遷移は、1) Operation の実行、2) event, または 3) 何らかの Functional Object 内部の活動、によってトリガされた際に発生する。

When a transition of a State Transition class of a State Machine occurs, the value of the State Attribute of the State Machine changes.

ある State Machine で、ある State Transition class の遷移が発生すると、その State Machine の State Attribute の値が変化する。

3.6.6.2. Trigger classes of State Transition classes

A State Transition class **shall** have one or more Trigger classes. A Trigger class **shall** be an Operation, an Event class, or Spontaneous.
- If a State Transition class has an Operation as a Trigger class, a transition of the State Transition class **shall** occur as a result of execution of the Operation.
- If a State Transition class has an Event class as a Trigger class, a transition of the State Transition class **shall** occur as a result of an event of the Event class.
- If a State Transition class has a Trigger class which is Spontaneous, a transition of the State Transition class occurs by some unidentified internal activity of the Functional Object.

State Transition class は、一つ以上の Trigger classes を持つこと。Trigger class は、Operation、Event class、または Spontaneous であること。
- ある State Transition class が、ある Operation を Trigger class として持つ場合、その Operation の実行の結果として、その State Transition class の遷移が発生すること。
- ある State Transition class が、ある Event class を Trigger class として持つ場合、その Event class の event の結果として、その State Transition class の遷移が発生すること。
- ある State Transition class が、Spontaneous である Trigger class を持つ場合、Functional Object 内部の何らかの未知の活動によって、その State Transition class の遷移が発生する。

A State of a State Machine **shall** be the Initial State or reachable from at least one of the other States in the State Machine. Hence, all the States are connected by one or more State Transition classes in a State Machine.

State Machine の State は、Initial State であるかその State Machine の他の少なくとも一つの States から遷移可能であること。したがって、State Machine では、全ての States が一つ以上の State Transition classes で接続されている。

3.6.6.3. Maximum Allowable Transition Time and Minimum Allowable Transition Time

For each State Transition class in a State Machine of a Functional Object, the Maximum Allowable Transition Time (the maximum time that a transition is allowed to take) **may** be specified. If a transition of a State Transition class invoked by an Operation is not completed within its Maximum Allowable Transition Time, the Functional Object **shall** be diagnosed by other entities as not functioning correctly.

ある Functional Object の State Machine の各 State Transition class について、Maximum Allowable Transition Time（遷移に許容される最大時間）を定めて**良い**。ある Operation によって呼び出された State Transition class の遷移がその Maximum Allowable Transition Time 内に完了しない場合、その Functional Object は、他の構成要素によって正常に機能していないと診断される**こと**。

Similarly, the Minimum Allowable Transition Time (the minimum time that a transition is allowed to take) **may** be specified for each type of State Transition class. If a transition of a State Transition class invoked by an Operation is completed in less than its Minimum Allowable Transition Time, the Functional Object **shall** be diagnosed by other entities as not functioning correctly.

同様に、Minimum Allowable Transition Time（遷移に許容される最小時間）を各 State Transition class に対して定めて**良い**。ある Operation によって呼び出された State Transition class の遷移がその Minimum Allowable Transition Time 未満で完了した場合、その Functional Object は、他の構成要素によって正常に機能していないと診断される**こと**。

3.6.6.4. Effective States for Operations

When a State Machine has State Transition classes which have an Operation as a Trigger classes, States referred to as Effective States for the Operation exist for the State Machine. Here, Effective States of a State Machine for an Operation are the States which are the Begin States of the State Transition classes that have the Operation as a Trigger class.

ある State Machine が、ある Operation を Trigger classes として持つ State Transition classes を持つ場合、その State Machine に対して、その Operation に対する Effective States と称する States が存在する。ここで、State Machine のある Operation に対する Effective States とは、その Operation を Trigger class として持つ State Transition classes の Begin States である States のことである。

3.7. DIAGNOSTIC RULES

A rule with which other entities diagnose whether a Functional Object is functioning correctly or not is referred to as a Diagnostic Rule. Whether a Functional Object is functioning correctly or not can be diagnosed by another entity, using a set of the Diagnostic Rules specified for the Functional Object. A Functional Object **shall** have zero or more Diagnostic Rules.

Functional Object が、正常に動作しているか否かを他の構成要素から診断するための規則を、Diagnostic Rule と称する。ある Functional Object に対して定めた一連の Diagnostic Rules を用い、その Functional Object が正常に機能しているか否かを他の構成要素によって診断できる。Functional Object は、ゼロ個以上の Diagnostic Rules を持つ**こと**。

A Diagnostic Rule **may** be specified as a Condition Expression. If the evaluation result of a Condition Expression which includes Attributes of a Functional Object is true, the Functional Object **shall** be diagnosed to be functioning normally, or else diagnosed to be functioning abnormally.

Diagnostic Rule は、条件式として定めて**良い**。ある Functional Object の Attribute を含む条件式を評価した結果が真であれば、その Functional Object は正常に機能していると診断される。そうでない場合、異常に機能していると診断される**こと**。

For each Diagnostic Rule, a text message containing additional information on the diagnosis (for example, (1) the level of abnormality and/or (2) methods to handle the abnormality) **shall** be specified.

各 Diagnostic Rule について、診断に関する追加情報（例えば、(1) 異常のレベルや、(2) 異常を処理する方法）を含む文字情報を定める**こと**。

The simplest Diagnostic Rule takes the form that specifies a pair of the boundary values for the allowed range for a Numerical Value Attribute of a Functional Object, with which other entities check whether the value of the Numerical Value Attribute is in or out of the allowed range (see Section 3.3.7).

最も単純な Diagnostic Rule は、ある Functional Object のある Numerical Value Attribute に対して許容範囲の境界値の組を定めるものである。これを用い、他の構成要素が、その Numerical Value Attribute の値が許容範囲に含まれるか否かをチェックする（3.3.7 項参照）。

[Note] When a Functional Object detects an anomaly in itself, it can report the anomaly to other entities by issuing an alert (see Section 3.5). If it is not easy for a Functional Object to detect an anomaly in itself, other entities diagnose the Functional Object, using the Diagnostic Rules specified for the Functional Object.

[注] ある Functional Object が自らの異常を検出した場合、alert を発行する事でその異常を他の構成要素に通知する事ができる（3.5 項参照）。ある Functional Object が自身で異常を検出する事が容易ではない場合、他の構成要素が、その Functional Object に定めた Diagnostic Rules を用い、その Functional Object を診断する。

3.8. OTHER FEATURES // その他

A Functional Object **may** generate data that are not specified as Attributes in the Functional Object definition as a result of executing its functions. For example, data representing results of observations or experiments (*e.g.*, images) **may** be generated by Functional Objects. Functional Objects **may** also reference the values of Attributes that other Functional Objects have.

ある Functional Object は、その機能を実行した結果として、その Functional Object の定義で Attributes として定めていないデータを生成しても**良い**。例えば、観察や実験の結果（画像等）を表すデータを、Functional Objects が生成しても**良い**。Functional Objects は、他の Functional Objects が有する Attributes 値を参照しても**良い**。

[Note] The Spacecraft Monitor and Control Protocol (SMCP) [R3] does not specify any methods for sending and receiving data that are not specified as Attributes.

[注] Spacecraft Monitor and Control Protocol (SMCP) [R3] は、Attributes として定めていないデータの送受信方法を定めない。

3.9. CONDITION EXPRESSION // 条件式

To specify a condition of various kinds (see Section 3.1), an expression referred to as a Condition Expression is used. A Condition Expression takes a boolean value of either true or false at a given time.

各種の条件（3.1 項参照）を定めるのに、条件式と称する式を用いる。条件式は、ある時点で、真または偽の何れかの真偽値を取る。

A Condition Expression consists of terms referred to as Comparison Terms and operators referred to as Logical Operators. The precise definition of Condition Expression is shown in List 3-1. A Condition Expression **shall** be one of a combination of "NOT" and a Condition Expression, a combination of "AND" and two or more Condition Expressions, a combination of "OR" and two or more Condition Expressions, a Comparison Term. Here, "NOT" is a unary Logical Operator and "AND" and "OR" are multi-term Logical Operators. A Comparison Term **shall** be a combination of an Attribute, an operator referred to as a Comparison Operator, and a constant, where a Comparison Operator **shall** be one of "equal to", "not equal to", "greater than", "greater than or equal to", "smaller than", and "smaller than or equal to".

条件式は、Logical Operators と称する演算子と Comparison Terms と称する項からなる。List 3-1 に条件式の正確な定義を示す。条件式は、"NOT" と条件式の組み合わせ、"AND" と二つ以上の条件式の組み合わせ、"OR" と二つ以上の条件式の組み合わせ、または Comparison Term の何れかであること。ここで、"NOT" は単項の Logical Operator であり、"AND" と "OR" は多項の Logical Operators である。また、Comparison Term は、Attribute, Comparison Operator と称する演算子，及び定数の組み合わせであること。ここで Comparison Operator は、「等しい」、「等しくない」、「より大きい」、「より大きいか等しい」、「より小さい」、及び「より小さいか等しい」の何れかであること。

Condition Expression ::=

Comparison Term | NOT Condition Expression |
AND Condition Expression Condition Expression + | OR Condition Expression Condition Expression +

Comparison Term ::=

Attribute Comparison Operator Constant

Comparison Operator ::=

equal to | not equal to | greater than | greater than | equal to | smaller than | smaller than | equal to

List 3-1 Extended BNF Definition of Condition Expression
拡張 BNF による条件式の定義

The left side of :: = indicates the item to be defined, and the right side indicates the content of the definition.
+ indicates the preceding term existing one or more times.
| is used to separate alternative terms.

::= の左辺は定義される項目、右辺は定義内容を示す。
+ は前項が一つ以上登場する事を示す。
| は前項・次項が選択肢である事を示す。

This notation is the extended BNF (*i.e.* BNF extended with regular expression) but does not specify the syntax.

この表記法は拡張 BNF（つまり、正規表現で拡張された BNF）であるが、構文は定めない。

4. FUNCTIONAL CLASS

4.1. GENERAL // 一般

If identical or similar Functional Objects are used in one or multiple spacecrafts and their onboard instruments, the common features of these Functional Objects can be specified as a template, which is referred to as a Functional Class. Conversely, a Functional Object can be specified with a Functional Class as a template.

同一または類似の Functional Objects を一つまたは複数の宇宙機やその搭載機器で用いる場合、これらの Functional Objects の共通の特徴を、Functional Class と称するひな型として定める事ができる。逆に言うと、Functional Object は、Functional Class をひな型として定める事ができる。

A Functional Class specifies design parameters and Attributes, Operations, Event classes, Alert classes, State Machines, and Diagnostic Rules that are used commonly by multiple Functional Objects.

Functional Class は、設計パラメータ、並びに、複数の Functional Objects で共通に用いる Attributes, Operations, Event classes、Alert classes 、State Machines 及び Diagnostic Rules を定める。

A Functional Object **shall** be specified by using zero or more Functional Classes as the templates and with or without additional functions (such as Attributes and/or Operations).

Functional Object は、ゼロ個以上の Functional Classes をひな型に用い、そのままか、何かの機能（Attributes や Operations 等）を追加して定めること。

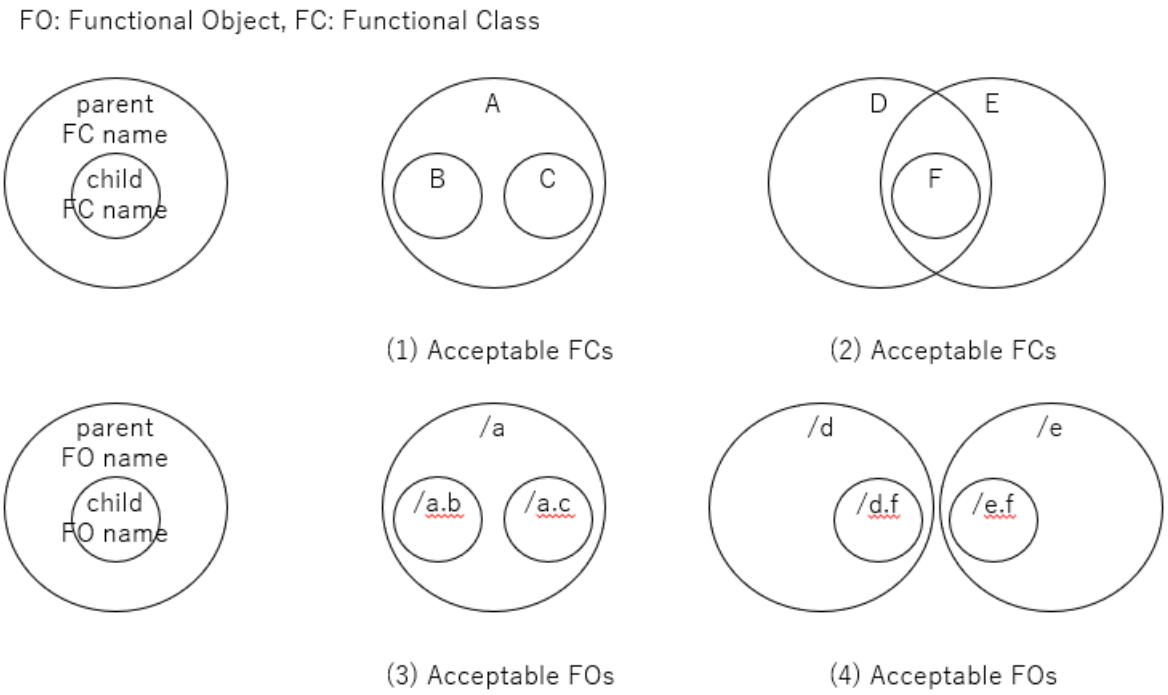

Figure 4-1 Acceptable Combination of Parent-Child Relation between Functional Classes/Functional Objects // Functional Classes/Functional Objects の親子関係で許容される組み合わせ

4.2. PARENT-CHILD RELATION // 親子関係

A Functional Class **shall** contain zero or more Functional Classes.

Functional Class は、Functional Classes をゼロ個以上含む**こと**。

[Example 1]
Figure 4-1 (1): Functional Class A contains Functional Class B and Functional Class C.
Figure 4-1 (3): "Functional Object /a specified with Functional Class A as the template" has "Functional Object /a.b specified with Functional Class B as the template" and "Functional Object /a.c specified with Functional Class C as the template".

[例 1]
Figure 4-1 (1): Functional Class A は、Functional Class B と Functional Class C を含む。
Figure 4-1 (4): 「Functional Class A をひな型として定めた Functional Object /a」は、「Functional Class B をひな型として定めた Functional Object /a.b」と「Functional Class C をひな型として定めた Functional Object /a.c」を持つ。

A Functional Class **may** be contained in two or more Functional Classes.

Functional Class は、二つ以上の Functional Classes に含まれて**良い**。

[Note] A Functional Object is not contained in two or more Functional Objects (see Section 3.2.1).

[注] Functional Object は、二つ以上の Functional Objects には含まれない（3.2.1 項参照）。

[Example 2]
Figure 4-1 (2): Functional Class D and Functional Class E both contain Functional Class F.
Figure 4-1 (4): "Functional Object /d specified with Functional Class D as the template" has "Functional Object /d.f specified with Functional Class F as the template". "Functional Object /e specified with Functional Class E as the template" has "Functional Object /e.f specified with Functional Class F as the template". Whereas both Functional Object /d.f and Functional Object /e.f are specified with Functional Class F as the template, Functional Object /d.f and Functional Object /e.f are different Functional Objects.

[例 2]
Figure 4-1 (2): Functional Class D と Functional Class E は、何れも、Functional Class F を含む。
Figure 4-1 (4): 「Functional Class D をひな型として定めた Functional Object /d」は、「Functional Class F をひな型として定めた Functional Object /d.f」を持つ。「Functional Class E をひな型として定めた Functional Object /e」は、「Functional Class F をひな型として定めた Functional Object /e.f」を持つ。Functional Object /d.f と Functional Object /e.f は何れも Functional Class F をひな型として定めたものであるが、Functional Object /d.f と Functional Object /e.f は異なる Functional Objects である。

4.3. STANDARD FUNCTIONAL CLASS // 標準的な FUNCTIONAL CLASS

This document specifies the following standard Functional Class:

1) Memory Functional Class (see Chapter 5).

本書では標準的な Functional Class として以下のものを定める。

1）Memory Functional Class (5 章参照)

5. MEMORY FUNCTIONAL CLASS

5.1. GENERAL // 一般

A Memory Functional Object is a Functional Object that represents a memory device (a device that stores data). The Memory Functional Class is an abstraction of the properties of all the Memory Functional Objects and specifies the design parameters and Operations that any Memory Functional Object **shall** or **may** have.

Memory Functional Object は、メモリデバイス（データを格納するデバイス）を表す Functional Object である。 Memory Functional Class は、全ての Memory Functional Objects が持つ性質を抽象化したものであり、Memory Functional Object が持た**なければならない**、または持って**良い**設計パラメータと Operations を定める。

The Memory Functional Object has the design parameters and Operations specified in this chapter. A Memory Functional Object **may** have (1) other Operations in addition to those specified below and (2) some other characteristics of Functional Objects (such as Attributes, Event classes and/or Alert classes).

Memory Functional Object は、本章に定める設計パラメータと Operations を持つ。Memory Functional Object は、(1) 以下に定める Operations に加え他の Operations, (2) その他の Functional Objects の特性（Attributes, Event classes や Alert classes 等）を持って**良い**。

5.2. DESIGN PARAMETERS // 設計パラメータ

5.2.1. General // 一般

A Memory Functional Object **shall** have the following design parameters. The values of these design parameters **shall** be specified and fixed at the time of the designing.

Memory Functional Object は、以下の設計パラメータを持つ**こと**。これらの設計パラメータの値は、Memory Functional Object の設計時に定め、固定される**こと**。

1) FirstAddress

2) LastAddress

3) MaximumUploadLength (optional)

4) AlignmentLength

1) FirstAddress

2) LastAddress

3) MaximumUploadLength（オプション）

4) AlignmentLength

5.2.2. FirstAddress and LastAddress

The FirstAddress and LastAddress **shall** specify, respectively, the first and last addresses of the memory area which can be accessed.

FirstAddress 及び LastAddress は、それぞれ、アクセスできるメモリ領域の最初及び最後のアドレスを定める**こと**。

5.2.3. MaximumUploadLength

The MaximumUploadLength **shall** specify the maximum octet length (*e.g.* 256 octets) of the memory data handled in one MemoryLoad Operation (see Section 5.3).

MaximumUploadLength は、一回の MemoryLoad Operation (5.3 項参照) が扱うメモリデータの最大 octet 長（例えば 256 octets）を定める**こと**。

5.2.4. AlignmentLength

The AlignmentLength **shall** specify the unit for memory data access, which **shall** be one of the following:
- 1 octet (*i.e.* any address),
- 2 octets,
- 4 octets, or
- MaximumUploadLength.

AlignmentLength は、メモリデータのアクセス単位を定める**こと**。また、以下の値の何れかである**こと**。
- 1 octet（つまり、任意のアドレス）、
- 2 octets,
- 4 octets, または
- MaximumUploadLength

5.3. OPERATIONS

5.3.1. General // 一般

A Memory Functional Object **may** have the following Operations. The Parameters for these Operations are given in parentheses.

1) MemoryLoad (StartAddress, MemoryData) (optional)
2) MemoryDump (NoOfDumps, StartAddress, Length) (optional)

Memory Functional Object は、次の Operations をもって**良い**。これらの Operations の Parameters は、（）内に示される。

1) MemoryLoad (StartAddress, MemoryData) (オプション)
2) MemoryDump (NoOfDumps, StartAddress, Length) (オプション)

5.3.2. MemoryLoad

The MemoryLoad is an Operation for uploading data to the memory. If the MemoryLoad is invoked, the Memory Functional Object **shall** write the value of the MemoryData into the memory area starting from the address specified by the StartAddress.

Note that the length of the MemoryData **shall** be equal to or smaller than the MaximumUploadLength, and the values
- the StartAddress and
- the octet lengths of the MemoryData

shall be multiples of the AlignmentLength.

[Note 1] The octet length of MemoryData is always the MaximumUploadLength if the AlignmentLength is equal to the MaximumUploadLength.

[Note 2] A telecommand message that invokes a MemoryLoad is called a MEMORY LOAD Telecommand in [R3].

MemoryLoad は、メモリにデータをアップロードするための Operation である。MemoryLoad が呼び出されると、Memory Functional Object は、MemoryData の値を、StartAddress が指定するアドレスから始まるメモリ領域に書き込む**こと**。

ここで、MemoryData の octet 長は MaximumUploadLength より小さいか等しい**こと**、また、
- StartAddress 及び
- MemoryData の octet 長

の値は、AlignmentLength の倍数である**こと**。

[注 1] AlignmentLength が MaximumUploadLength に等しい場合、MemoryData の octet 長は常に MaximumUploadLength である。

[注 2] MemoryLoad を呼び出す telecommand message を、[R3] では、MEMORY LOAD Telecommand と呼ぶ。

5.3.3. MemoryDump

The MemoryDump is an Operation for dumping memory data. If the MemoryDump is invoked, the Memory Functional Object **shall** dump the memory data of the octet length specified by the Length, starting at the address specified by the StartAddress, for the number of times specified by the NoOfDumps.

MemoryDump は、メモリデータをダンプ（読出し）するための Operation である。MemoryDump が呼び出されると、Memory Functional Object は、NoOfDumps で指定された回数、StartAddress で指定されたアドレスから、Length で指定された octet 長のメモリデータをダンプすること。

[Note 1] A telecommand message that invokes a MemoryDump is called a MEMORY DUMP Telecommand in [R3].

[注 1] MemoryDump を呼び出す telecommand message を、[R3] では、MEMORY DUMP Telecommand と呼ぶ。

[Note 2] A telemetry message which transfers memory data is called a MEMORY DUMP Telemetry in [R3].

[注 2] メモリデータを伝送する telemetry message を、[R3] では、MEMORY DUMP Telemetry と呼ぶ。

A Memory Functional Object **should** be able to dump the entire memory area of the Memory Functional Object with an invocation of the MemoryDump; [Rationale] Because GSTOS has a function that collates whether the contents of a memory are as expected through the MEMORY DUMP Telemetries generated by one MEMORY DUMP Telecommand and notifies the operator of the collation result and this function enables a simple spacecraft operation.

Memory Functional Object は、1 回の MemoryDump の呼び出しで、Memory Functional Object の全メモリ領域をダンプ可能であるべきである； [根拠] GSTOS は、1 つの MEMORY DUMP Telecommand により生成された MEMORY DUMP Telemetries を通じてメモリが期待された値か照合し、その照合結果を操作者に知らせる機能を持っている。この機能を用いると単純な宇宙機運用が実現できるため。

Appendix A. Acronyms // 略語集

This chapter lists the acronyms used in this document.

本章では、本書が用いる略語一覧を示す。

GSTOS	Generic Spacecraft Test and Operations Software
NOP	No Operation
SIB	Spacecraft Information Base
SMCP	Spacecraft Monitor & Control Protocol
XOR	Exclusive OR

Appendix B. An Example of a Functional Object

B.1. GENERAL // 一般

In this Appendix, a Functional Object named X_A is presented as an example of a Functional Object. This Functional Object models the basic functions to control a simple instrument.

ここでは、Functional Object の例として、X_A という Functional Object について説明する。この Functional Object は、単純な機器の実行を制御する基本的な機能をモデル化したものである。

B.2. FUNCTIONAL OBJECT

A Functional Object named X_SubSystem represents a Sub-System X.

X_SubSystem と命名された Functional Object はサブシステム X を示す。

Functional Object X_A is contained in the parent Functional Object X_SubSystem and specifies the functions of the instrument X-a contained in Sub-System X.

Functional Object X_A は、親 Functional Object X_SubSystem に含まれており、サブシステム X 内の X-a という機器の機能を定める。

X_A contains children Functional Objects named X_A1 and X_A2, each of which represents a set of functions performed by the instrument X-a. X_A specifies the functions concerning the entire instrument.

X_A は、X_A1 と X_A2 と命名された子 Functional Objects を含み、そのそれぞれは機器 X-a で実行される機能の集合を表わす。X_A は、機器全体に関する機能を定める。

X_A is valid only when certain Attributes of the parent Functional Object X_SubSystem meet a certain condition. The children Functional Objects X_A1 and X_A2 are valid only when the value of the State Attribute X_A_RunStop (see Section B.6) is RUN, *i.e.*, X_A is in the State "RUN".

X_A は、親 Functional Object X_SubSystem の特定の Attributes が特定の条件を満たすときのみ有効となる。子 Functional Objects X_A1 と X_A2 は、State Attribute X_A_RunStop（B.6 項参照）の値が RUN，つまり、X_A が State "RUN"にあるときのみ有効である。

X_A has Attributes and Operations for monitoring and controlling the instrument X-a as a whole. It has a State Machines that represent the rules concerning its actions. It also has an Event class, for which an Alert class is defined.

X_A は、機器 X-a 全体を監視制御するための Attributes と Operations を有している。これは動作規則に対応する State Machines を有する。さらに、一つの Event class を持ち、これに対して一つの Alert class が定義される。

B.3. ATTRIBUTES

X_A has the following four Attributes:

1) X_A_OnOff,
2) X_A_RunStop,
3) X_A_ErrorStatus, and
4) X_A_CheckMode.

The values of X_A_OnOff and X_A_RunStop represent the States in which the Functional Object is at a given time. Thus, they are State Attributes. The other Attributes, X_A_ErrorStatus and X_A_CheckMode, are Numerical Value Attributes.

Of the four Attributes, only the value of X_A_CheckMode can be set from the outside of the Functional Object.

X_A_OnOff is valid whenever the Functional Object is valid, whereas X_A_RunStop, X_A_ErrorStatus, and X_A_CheckMode are valid only when the value of X_A_OnOff is ON, *i.e.*, the Functional Object is in the State "ON" (see Section B.6).

X_A は、次の四つの Attributes を持つ。

1) X_A_OnOff
2) X_A_RunStop
3) X_A_ErrorStatus
4) X_A_CheckMode

X_A_OnOff と X_A_RunStop の値は、この Functional Object のある時点の States を表す。したがって、これらは State Attributes である。その他の Attributes、すなわち X_A_ErrorStatus と X_A_CheckMode は、Numerical Value Attributes である。

これら四つの Attributes の中で、X_A_CheckMode のみが Functional Object 外部から値を設定できる。

X_A_OnOff は、Functional Object が有効な時は常に有効である。一方、X_A_RunStop, X_A_ErrorStatus, 及び X_A_CheckMode は X_A_OnOff の値が ON, つまり、この Functional Object が State "ON", にある場合のみ有効である（B.6 項参照）。

B.4. OPERATIONS

X_A has the following five Operations:

1) X_A_On,
2) X_A_Start,
3) X_A_Stop,
4) X_A_Off, and
5) X_A_SetCheckMode.

The Operation X_A_SetCheckMode holds a value which is set to the Numerical Value Attribute X_A_CheckMode.

These Operations can be executed when a set of the following conditions are met. These Operations also appear as the Trigger classes of the State Transition classes of the State Machine (see Figure B-1).

1) X_A_On X_A_OnOff=OFF
2) X_A_Start X_A_OnOff=ON AND X_A_RunStop=STOP
3) X_A_Stop X_A_OnOff=ON AND X_A_RunStop=RUN
4) X_A_Off X_A_OnOff=ON
5) X_A_SetCheckMode X_A_OnOff=ON

X_A は、次の五つの Operations を持つ。

1) X_A_On
2) X_A_Start
3) X_A_Stop
4) X_A_Off
5) X_A_SetCheckMode

Operation X_A_SetCheckMode は、Numerical Value Attribute X_A_CheckMode に値を設定する値を保持する。

これらの Operations は、以下の条件が満たされたときに実行できる。また、これらの Operations は、State Machine の State Transition classes の Trigger classes としても登場する（Figure B-1 参照）。

B.5. EVENT CLASSES

X_A has the following single Event class:

1) X_A_ErrorDetect.

An event of this Event class is triggered when the following condition is met:

NOT (X_A_ErrorStatus=NORMAL)

X_A can issue alerts of the following single Alert class:

1) X_A_ErrorDetected.

An alert of this Alert class is used to report to other entities the occurrence of an event of the Event class X_A_ErrorDetect and has the value of the Attribute X_A_ErrorStatus as a Parameter.

X_A は、以下の一つの Event class のみを持つ。

1) X_A_ErrorDetect.

この Event class の event は、次の条件が満たされた場合にトリガされる。

NOT (X_A_ErrorStatus=NORMAL)

X_A は、次の一つの Alert class の alerts を発行できる。

1) X_A_ErrorDetected.

この Alert class の alert は、Event class X_A_ErrorDetect の event の発生を他の構成要素に通知するために用いられ、Attribute X_A_ErrorStatus の値を Parameter として持つ。

B.6. STATE MACHINES

The behavior of X_A is specified with two State Machines, X_A_OnOff and X_A_RunStop (see Figure B-1).

X_A の振る舞いは、二つの State Machines X_A_OnOff と X_A_RunStop で定める（Figure B-1 参照）。

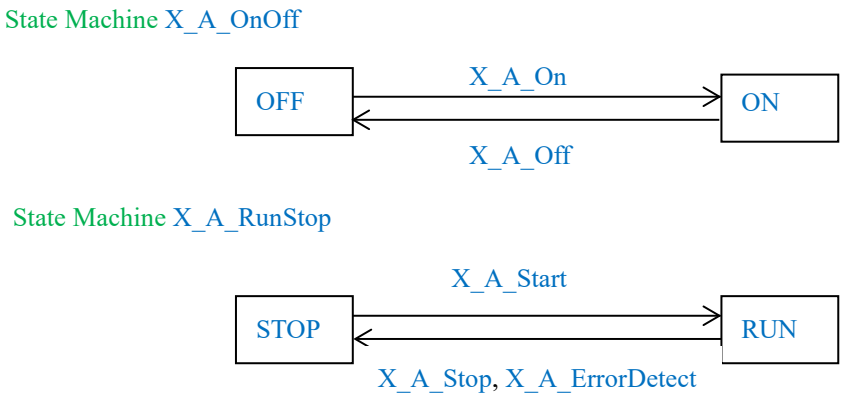

Figure B-1: State Machines of Functional Object X_A

The boxes show the States. The arrows show the directions of a transition of the State Transition classes. The Trigger classes are shown above the arrows. // 四角は、States を示す。矢印は、State Transition classes の遷移方向を示す。矢印の上に Trigger classes を示す。

The State Machine X_A_OnOff takes either of the two States "OFF" and "ON". The Initial State is OFF. The Current State of this State Machine is indicated by the State Attribute X_A_OnOff. This State Machine is valid whenever the Functional Object is valid.

State Machine X_A_OnOff は、二つの States "OFF" と "ON" のいずれかを取る。Initial State は OFF である。この State Machine の Current State は、State Attribute X_A_OnOff で示される。この State Machine は、Functional Object が有効な場合は常に有効である。

The State Machine X_A_RunStop takes either of the two States "STOP" and "RUN". The Initial State is STOP. The Current State of this State Machine is indicated by the State Attribute X_A_RunStop. This State Machine is valid only when the value of X_A_OnOff is ON (*i.e.*, when the Functional Object is in the State "ON").

State Machine X_A_RunStop には、二つの States "STOP" と "RUN" がある。Initial State は STOP である。この State Machine の Current State は、State Attribute X_A_RunStop で示される。この State Machine は、X_A_OnOff の値が ON の場合（つまり、Functional Object が State ON の場合）にのみ有効である。

Appendix C. History of Terminology Changes
用語の変更の履歴

NOTICE-1:
Operation • From: Attributes of Operations • To: Value Setting Attributes of Operations Alert class • From Attributes of Alerts classes • To: Value Notifying Attributes of Alerts classes