MASTER'S THESIS

Comparison of the CCSDS Mission Operations Services with the Packet Utilization Standard Services

César Coelho
2014

Master of Science (120 credits)
Space Engineering - Space Master

Luleå University of Technology
Department of Computer Science, Electrical and Space Engineering
Master Thesis

Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

by

César Coelho

Joint European Master in Space Science and Technology

European Space Agency Supervisor: Dr. Mehran Sarkarati
Luleå University of Technology Supervisor: Dr. Johnny Ejemalm

September, 2014
Table of contents

1 ABSTRACT .............................................................................................................................. 3
2 INTRODUCTION ................................................................................................................... 4
  2.1 General ............................................................................................................................... 4
  2.2 CCSDS Mission Operations Services ............................................................................. 6
  2.3 Acronyms .......................................................................................................................... 7
3 PUS – CCSDS MO SERVICES COMPARISON ................................................................. 8
  3.1 Conventions ....................................................................................................................... 8
  3.2.1 Procedure .................................................................................................................... 10
  3.2.2 PUS Service Concept ................................................................................................. 10
  3.2.3 PUS: Requests and Reports ....................................................................................... 12
  3.2.4 PUS-C: Requests and Reports ................................................................................... 33
  3.3 Comparison Summary ...................................................................................................... 46
  3.3.1 [1] Telecommand Verification Service (COM: Activity) ............................................. 46
  3.3.2 [3] Housekeeping and Diagnostic Data Reporting Service (M&C: Aggregation) .... 47
  3.3.3 [4] Parameter Statistics Reporting Service (M&C: Statistic) .................................... 49
  3.3.4 [5] Event Reporting Service (M&C: Alert) ................................................................. 50
  3.3.5 [12] On-board Monitoring Service (M&C: Check) .................................................... 51
  3.3.6 [20] On-board Parameter Management Service (M&C: Parameter) ......................... 55
  3.3.7 Others .......................................................................................................................... 56
  3.4 Review Item Disposition (RID) ....................................................................................... 57
    3.4.1 RID 01 ......................................................................................................................... 57
    3.4.2 RID 02 ......................................................................................................................... 58
    3.4.3 RID 04 ......................................................................................................................... 59
    3.4.4 RID 05 ......................................................................................................................... 60
    3.4.5 RID 06 ......................................................................................................................... 61
    3.4.6 RID 07 ......................................................................................................................... 62
    3.4.7 RID 08 ......................................................................................................................... 63
    3.4.8 RID 09 ......................................................................................................................... 64
    3.4.9 RID 10 ......................................................................................................................... 65
    3.4.10 RID 12 ....................................................................................................................... 66
    3.4.11 RID 13 ....................................................................................................................... 67
    3.4.12 RIDs feedback .......................................................................................................... 68
4 IMPLEMENTATION OF 2 CCSDS MO SERVICES ON A MITYARM BOARD ............. 70
  4.1 Introduction ....................................................................................................................... 70
  4.2 MITYARM-5CSX and OPS-SAT relation ...................................................................... 70
  4.3 Software Architecture: General Overview ...................................................................... 71
  4.4 Provider side - CLI .......................................................................................................... 72
    4.4.1 Parameter Service ....................................................................................................... 72
    4.4.2 Aggregation Service .................................................................................................. 73
    4.4.3 Parameter Manager ................................................................................................. 73
    4.4.4 Aggregation Manager ............................................................................................... 73
    4.4.5 Monitoring Parameters ............................................................................................ 73
    4.4.6 Instruments Simulator .............................................................................................. 73
  4.5 Consumer side - GUI ....................................................................................................... 74
  4.6 Mity USB Terminal .......................................................................................................... 76
5 CONCLUSIONS ...................................................................................................................... 77
6 ACKNOWLEDGEMENTS ......................................................................................................... 78
1 ABSTRACT

The Master Thesis proposes the development of a CCSDS Mission Operations (MO) comparison with the Packet Utilization Standard (PUS) by doing a systematic mapping of all the PUS “telecommand packet, application data” and “telemetry source packet, source data” into the equivalent MO operations and objects. The main advantage of using this systematic approach is the obtainment of an extensive mapping of all the current PUS/MO equivalent services in order to find the main comparison summary of the strengths and weaknesses of CCSDS MO services when juxtaposed with PUS.

Currently, the European Cooperation for Space Standardisation (ECSS) is developing the next PUS, issue C (PUS-C). In this thesis, the systematic mapping approach was also performed on the latest PUS-C draft book (from 11 June 2014).

After making the extensive systematic mapping, it was possible to produce a concise comparison summary which identifies previous unknown deficiencies of CCSDS MO when put in contrast with PUS and PUS-C. These were reported back as Review Item Disposition (RIDs) in order to enhance the next draft of the Standard.

As a secondary thesis task, there was an implementation of the CCSDS Mission Operations Monitor and Control Aggregation Service and Parameter Service on a MityARM-5CSX processor card as a Proof of Concept Prototype for future Space Missions. The MityARM-5CSX processor card is a highly configurable System-on-Module that features an Altera Cyclone V capable of running real-time Operating Systems. This state-of-the-art product shall be incorporated on-board of the future OPS-SAT mission.
2 INTRODUCTION

2.1 General

This Master Thesis was developed during an internship at the European Space Operations Centre (ESOC) in Darmstadt in the Mission Data Systems - Applications and Special Projects Data Systems Section from March 2014 until August 2014.

The Consultative Committee for Space Data Systems (CCSDS) is currently developing a new set of Standards for future Space Missions which are based on a Service Oriented Architecture. These new Standards are known as the Mission Operation (MO) services and they provide a set of services on-board of a spacecraft which can be used and operated by a remote consumer.\(^1\)

At the current time, the Mission Operations: Monitor and Control services (MO M&C) are under agency review and they have a key role as it provides the most basic services for monitoring and controlling various system elements.\(^2\)

The Mission Operations: Monitor and Control services currently in development by CCSDS provide equivalent services of the Packet Utilization Standard (PUS) services. PUS was developed by the European Corporation for Space Standardisation (ECSS) and it is the current Standard in use for telemetry and telecommands on all ESA spacecraft.

![Figure 1: Visual representation of the comparison between CCSDS MO and PUS](image)

Parallel to the MO development, there is also undergoing an upgrade of the PUS which is currently being developed by the ECSS. This upgrade is named Packet Utilization Standard issue C (PUS-C) and it is planned to be implemented in the near future providing an expansion and update for the current PUS issue A in use. The PUS-C new services and subservices were also compared with CCSDS MO services following the same exact procedures used for the PUS issue A telemetry and telecommand packets.

![Figure 2: PUS-C - the next PUS issue which will replace the current one](image)

As a new and more advanced technological standard, it is expected from the CCSDS Mission Operations services to completely cover all the PUS functionalities, to improve them and even expand them. In general terms, this thesis compares the services provided in the Mission Operations: Monitor and Control services with their corresponding equivalent Packet Utilization Standard services. Currently, it is known that the M&C still does not completely cover all their equivalent PUS functionalities, hence, one of the objectives of the internship was to discover to which extend and to make improvement suggestions after finding the

---

\(^1\) “CCSDS Mission Operations Services are Getting Real!” – Presentation, CCSDS SM&C Working Group and M. Merri

\(^2\) CCSDS 520.0-G-3 - Mission Operations Services Concept, December 2010
weaknesses. The final goal of covering all their equivalent services can be achieved by introducing some changes in the MO M&C standard which are present in this thesis on the Review Item Disposition (RIDs) section.

This thesis describes the procedure taken for the development of the comparison summary by presenting the systematic mapping of one of the services, the Aggregation Service, with the PUS service 3, Housekeeping and Diagnostic Data Reporting Service. After, the thesis contains the full summary which resulted from executing the same systematic mapping for all the CCSDS MO possible services. The systematic mapping for the remaining services is not presented in this thesis for two reasons, first, the procedure taken is the same as the one used for the Aggregation service, and second, due to their extend. Although, it is possible to consult them from ESA’s document: “Systematic Comparison of the Mission Operations Monitor and Control services with the Packet Utilization Standard services”. This document has the DMS reference: DHSO-MC-TN-1001-HSO-GDA and can be accessed from the reference on the footnote, [3]

The document mentioned above, contains the complete mapping of the corresponding PUS/MO services up to the moment, the comparison summary and the RIDs. During its development, there was on cycle of the iterative process of submitting Review Item Dispositions (RIDs) for the Mission Operations: Monitor & Control services book which allows any person to submit suggestions to review the document. This process allowed the MO M&C not only to become PUS compliant but also more robust. It is important to mention that the Mission Operations: Common Object Model (COM) Activity service even though it does not belong to the MO M&C book, it is also covered by the comparison.

The main objective of this thesis is to demonstrate that the MO M&C services can virtually cover all their PUS equivalent services. As a secondary objective, the Systematic mapping can also be used to facilitate the implementation of a MO to/from PUS adapter such as the one that is planned to be developed for the transitional architecture.

The following table shows which services and sub-services from PUS and PUS-C were compared with their equivalent CCSDS MO services:

<table>
<thead>
<tr>
<th>PUS service</th>
<th>CCSDS MO service</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1] Telecommand Verification Service</td>
<td>COM – Activity</td>
</tr>
<tr>
<td>[3] Housekeeping and Diagnostic Data Reporting Service</td>
<td>M&amp;C – Aggregation</td>
</tr>
<tr>
<td>[12] On-Board Monitoring Service</td>
<td>M&amp;C – Check</td>
</tr>
</tbody>
</table>

The comparison is mainly focused on the services from MO M&C, although the Activity Service from the MO COM was also included. It is also important to mention that PUS service 20, On-Board Parameter Service only exists in PUS-C.

During the internship at ESOC, an optional task project was also developed, the implementation of the CCSDS Mission Operations Monitor and Control: Aggregation Service and Parameter Service in a MityARM-5CSX as a Proof of Concept Prototype for future Space Missions. This project provides both a consumer side application and a provider side application. The provider offers the possibility to interact with parameters originated from a Magnetometer and a GPS flying in a simulated spacecraft in a dawn-dusk orbit at an altitude of around 650 kilometers (the expected orbit for the OPS-SAT mission). The MityARM-5CSX processor card is a highly configurable System-on-Module that features an Altera Cyclone V capable of running real-time Operating Systems. This state-of-the-art product shall be incorporated on-board of the future OPS-SAT spacecraft.

---

3 Systematic Comparison of the Mission Operations Monitor and Control services with the Packet Utilization Standard services, DHSO-MC-TN-1001-HSO-GDA, C. Coelho
2.2 CCSDS Mission Operations Services

The CCSDS Mission Operations services are a new set of Standards for future Space Missions based on a Service Oriented Architecture. These new standards provide a set of services on-board of a spacecraft which can be used and operated by a remote consumer. This service oriented framework aims at increasing the interoperability of mission operation services and to decrease the necessary integration effort for uniting the services from all the different stakeholders.\textsuperscript{[4]}

CCSDS MO uses a layered approach in order to reduce the implementation complexity. This approach allows the development of components which can be easily reused and it also allows the separation from the underlying implementation technology. The CCSDS Mission Operations services define an extensible set of end-to-end services which interacts between distributable mission operations functions such as software applications associated to the mission operations domain.\textsuperscript{[5]}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{mission_operations_service_framework.png}
\caption{Overview of the Mission Operations Service Framework}
\end{figure}

Currently, under agency review, the Monitor and Control service (M&C) has a key role as it provides the most basic services for monitoring and controlling various system elements. M&C includes services such as Action, Parameter, Alert, Aggregation, Check, and others.

\begin{itemize}
\item \textsuperscript{4} Intrinsic Interoperability of Services, M. Sarkarati, M. Merri, M. Spada – SpaceOps 2012
\item \textsuperscript{5} CCSDS 520.0-G-3 – Mission Operations Services Concept, December 2010
\end{itemize}
2.3 Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSDS</td>
<td>Consultative Committee for Space Data Systems</td>
</tr>
<tr>
<td>COM</td>
<td>Common Object Model</td>
</tr>
<tr>
<td>ECSS</td>
<td>European Corporation for Space Standardisation</td>
</tr>
<tr>
<td>ESA</td>
<td>European Space Agency</td>
</tr>
<tr>
<td>ESOC</td>
<td>European Space Operations Centre</td>
</tr>
<tr>
<td>MAL</td>
<td>Message Abstraction Layer</td>
</tr>
<tr>
<td>MO</td>
<td>Mission Operations</td>
</tr>
<tr>
<td>M&amp;C</td>
<td>Monitor and Control</td>
</tr>
<tr>
<td>OPS-SAT</td>
<td>First ESA’s CubeSat mission</td>
</tr>
<tr>
<td>PUS</td>
<td>Packet Utilization Standard</td>
</tr>
<tr>
<td>PUS-C</td>
<td>Packet Utilization Standard (issue C)</td>
</tr>
<tr>
<td>RID</td>
<td>Review Item Disposition</td>
</tr>
</tbody>
</table>
3 PUS – CCSDS MO SERVICES COMPARISON

Since the comparison resulted in a very extensive document due to its vast detail, the thesis will only cover one of the systematic mappings between the Aggregation Service and the PUS service 3, Housekeeping and Diagnostic Data Reporting Service, in order to give the reader of this thesis the necessary understanding of the systematic procedure used in the development of the comparison.

Nevertheless, as already mentioned before on the introduction, the full document has the following DMS reference: “DHSO-MC-TN-1001-HSO-GDA” and can be consulted via the footnote reference on this page. [6] Some conventions were defined to increase the readability experience and make it easier to follow. These conventions will be presented on the following section.

3.1 Conventions

3.1.1 Color Code

The color code used in this document increases the clarity for the reader and it will make it dramatically easier to follow.

All the variables relative to a packet within the Packet Utilization Standard book are represented in Bold Green. In the PUS Packet Table Representation, the first row will not have the Color Code enforced because it is implicitly assumed that it is referencing its PUS packet field name.

Relatively to MO, all its operations will be represented in Bold Blue. Its objects will be represented in Bold Black while the variables of these objects will be represented in Blue (no bold). In the MO Object Table Representation, the variables in the Field column will not have the Color Code enforced because it is implicitly assumed that the Field column is referencing to a particular MO object structure.

Constant values and values emerging due to specific requirement conditions will be represented in Bold Orange.

Relatively to the MO COM Archive service, all its operations will be represented in Bold Red. These shall override the previous Bold Blue Color Code convention.

Some simple functions are used in this document and they will be represented in Bold Purple.

Errors or limitations found in PUS and MO will be represented in Red Highlight. These errors will be addressed in the Comparison Summary.

3.1.2 Tables

In this document, some tables will be used in order to present some data in an orderly fashion manner and these will be used several times throughout this document without any caption. These tables will represent packets, objects or operation sequences.

To represent a PUS (issue A) packet, it shall be used the format in Table 3-1.

The dashed lines (appearing as straight lines after generating the file in pdf format) indicate that the packet is not complete and it continues on the next lines.

---

Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

Table 3-1: Example: PUS Packet Table Representation

<table>
<thead>
<tr>
<th>Field1</th>
<th>Field2</th>
<th>Field3</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt; FieldValue1 &gt;&gt;</td>
<td>&lt;&lt; FieldValue2 &gt;&gt;</td>
<td>&lt;&lt; FieldValue3 &gt;&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field4</th>
<th>Field5</th>
<th>Field6</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt; FieldValue4 &gt;&gt;</td>
<td>&lt;&lt; FieldValue5 &gt;&gt;</td>
<td>&lt;&lt; FieldValue6 &gt;&gt;</td>
</tr>
</tbody>
</table>

To represent a MO object, it shall be used the format in Table 3-2. If the represented MO object contains another object inside, this second object will be represented in **Bold** (e.g. << object1 >>) and the fields within this object shall be slightly shifted to the right (e.g. << field1InObject1 >>).

Table 3-2: Example: MO Object Table Representation

<table>
<thead>
<tr>
<th>&lt;&lt; ObjectType &gt;&gt;</th>
<th>&lt;&lt; ObjectName &gt;&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt; field1 &gt;&gt;</td>
<td>&lt;&lt; VariableType1 &gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt; field2 &gt;&gt;</td>
<td>&lt;&lt; VariableType2 &gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt; object1 &gt;&gt;</td>
<td>&lt;&lt; ObjectType1 &gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt; field1InObject1 &gt;&gt;</td>
<td>&lt;&lt; VariableType3 &gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt; field2InObject1 &gt;&gt;</td>
<td>&lt;&lt; VariableType4 &gt;&gt;</td>
</tr>
</tbody>
</table>

To represent a MO operations sequence, it shall be used the format in Table 3-3. The sequence of operations is sorted from left to write. So, the leftmost operation in the table representation shall be the first to be executed. The table representation is always preceded by its corresponding textual version. The table representation might be used to avoid ambiguities which might arise from the use of its textual version. If << OutputType1 >> has the same type as << InputType2 >>, then the output of the operation1 shall be the input of operation2.

Table 3-3: Example: MO Operations Sequence Table Representation

<table>
<thead>
<tr>
<th>Operation</th>
<th>operation1</th>
<th>operation2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>&lt;&lt;InputType1&gt;&gt;</td>
<td>&lt;&lt;InputType2&gt;&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;&lt;InputData1[SIZE]&gt;&gt;</td>
<td>&lt;&lt;InputData2[SIZE]&gt;&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output type</th>
<th>&lt;&lt;OutputType1&gt;&gt;</th>
</tr>
</thead>
</table>

(Both lists have a size <<SIZE>>)
Table 3-4: Example: PUS-C Packet Table Representation

<table>
<thead>
<tr>
<th>field1</th>
<th>field2</th>
<th>field3</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt;FieldValue1 &gt;&gt;</td>
<td>&lt;&lt;FieldValue2 &gt;&gt;</td>
<td>&lt;&lt;FieldValue3 &gt;&gt;</td>
</tr>
</tbody>
</table>

optional          optional

3.2 Systematic Mapping: Housekeeping and Diagnostic Data Reporting Service [3]

3.2.1 Procedure

The service 3, Housekeeping and Diagnostic data reporting service, from the Packet Utilization Standard (PUS) and the service Aggregation from the Mission Operations Monitoring and Control (MO M&C) will both be compared with each other in the direction PUS packet to MO M&C and also from MO M&C to PUS packet. This means that each packet of the PUS will be juxtaposed with the respective M&C object and operation in an attempt to match an equivalency.

Both services have the scope of providing for the reporting of all the information of operational significance that is not explicitly provided within the reports of another service.

The PUS service 3 is composed by two independent sub-services: Housekeeping; Diagnostic. These shall be represented in the Aggregation service (M&C) by setting the category field value in the AggregationDefinition structure to “GENERAL” (1) for Housekeeping:

\[
\text{AggregationDefinition.category} = \text{GENERAL}
\]

And by defining the field “category” in the AggregationDefinition structure as “DIAGNOSTIC” (2) for Diagnostic:

\[
\text{AggregationDefinition.category} = \text{DIAGNOSTIC}
\]

3.2.2 PUS Service Concept

3.2.2.1 General

In PUS there is a unique structure identification (SID) associated with each distinct reporting definition. For M&C, there are also different “name” fields for each definition which means that it is possible to set these with the same value as the PUS SID:

\[
\text{AggregationDefinition.name} = \text{int2strAgg(SID)}
\]

In the PUS, every telemetry packet is stamped with the SID in order to determine the nature of the packet by the ground station. In M&C Aggregation service the object AggregationValue is a COM object with the related link indicating the AggregationDefinition which caused its creation which means that the object is already linked to the aggregation definition that generated its creation and there is no need to include the SID in the structure once again.
3.2.2.2 Data collection

In PUS, each reporting definition has an associated data collection interval, which is the time interval over which the parameters are sampled. In M&C, this interval shall be defined by changing the updateInterval field in the `AggregationDefinition` structure:

\[
\text{AggregationDefinition.updateInterval} = \text{Collection Interval} \times \text{<DIAG_MIN_INTERVAL>}
\]

In M&C, an updateInterval field set to 0 means a non-periodic reporting.

3.2.2.3 Parameter report generation

In PUS, two modes of generating parameter reports exist:

a. Periodic Mode

In M&C, the aggregation definition of this mode shall be represented as having an “updateInterval” field in the `AggregationDefinition` different than zero and by having the filterEnabled field set to FALSE:

- `AggregationDefinition.updateInterval` != 0
- `AggregationDefinition.filterEnabled` == FALSE

In M&C, the aggregation shall be represented as having the “generationMode” field in the `AggregationValue` set to PERIODIC (enumerated as 2):

- `AggregationValue.generationMode` == PERIODIC
- `AggregationValue.filtered` == FALSE

b. Filtered Mode

In M&C, the aggregation definition of this mode shall be represented as having the `filterEnabled` field set to TRUE and the filtered field as TRUE in the `AggregationValue` structure:

- `AggregationDefinition.filterEnabled` == TRUE
- `AggregationValue.filtered` == TRUE

b1. When parameter changes exceeded some threshold.

In M&C, this mode corresponds to the PERIODIC `generationMode` (enumerated as 2), where the aggregation value is generated because of the filter detection:

- `AggregationValue.generationMode` == PERIODIC

b2. When the packet is generated due to timeout

In M&C, the aggregation shall have a `generationMode` field in the `AggregationValue` set to FILTERED_TIMEOUT (enumerated as 3):

- `AggregationValue.generationMode` == FILTERED_TIMEOUT

Parameter sampling times:

Absolute sampling times of parameters in housekeeping parameter reports shall be determinable to a given accuracy on the ground.

There are three alternative ways of achieving this requirement:

a. Knowledge of the on-board parameter sampling mechanism.

b. Time-stamped telemetry parameters.

c. Report sampling time offsets.

In PUS, the third alternative was chosen. It uses the requests and reports 13, 14, 15 and 16 to determine the parameter sampling time offsets. In M&C, the time offsets are stamped in every telemetry parameter (case b.) and are stored in the `deltaTime` field of the `AggregationValue`:

- `AggregationValue.deltaTime` = Sampling-Time Offset
### 3.2.3 PUS: Requests and Reports

In this section each one of the Requests and Reports from the PUS service will be mapped with its identical M&C operation and also the respective object data fields to be used, shall be presented. A summary table for PUS follows:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>M&amp;C Aggregation operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Define New Housekeeping Parameter Report</td>
<td>addDefinition()</td>
</tr>
<tr>
<td>2</td>
<td>(TC) Define New Diagnostic Parameter Report</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Clearing Housekeeping Parameter Report Definitions</td>
<td>removeDefinition()</td>
</tr>
<tr>
<td>4</td>
<td>(TC) Clearing Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(TC) Enable Housekeeping Parameter Report Generation</td>
<td>enableGeneration()</td>
</tr>
<tr>
<td>6</td>
<td>(TC) Disable Housekeeping Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(TC) Enable Diagnostic Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(TC) Disable Diagnostic Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(TC) Report Housekeeping Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(TC) Report Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(TM) Housekeeping Parameter Report Definitions Report</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(TM) Diagnostic Parameter Report Definitions Report</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(TC) Report Housekeeping Parameter Sampling-Time Offsets</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(TC) Report Diagnostic Parameter Sampling-Time Offsets</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(TM) Housekeeping Parameter Sampling-Time Offsets Report</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(TM) Diagnostic Parameter Sampling-Time Offsets Report</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(TC) Select Periodic Housekeeping Parameter Report Generation Mode</td>
<td>enableFilter()</td>
</tr>
<tr>
<td>18</td>
<td>(TC) Select Periodic Diagnostic Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>(TC) Select Filtered Housekeeping Parameter Report Generation Mode</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>20</td>
<td>(TC) Select Filtered Diagnostic Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>(TC) Report Unfiltered Housekeeping Parameters</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>(TC) Report Unfiltered Diagnostic Parameters</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>(TM) Unfiltered Housekeeping Parameters Report</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>(TM) Unfiltered Diagnostic Parameters Report</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>(TM) Housekeeping Parameters Report</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>(TM) Diagnostic Parameters Report</td>
<td></td>
</tr>
</tbody>
</table>

#### 3.2.3.1 [3,1] and [3,2] - (TC) Define New Housekeeping/Diagnostic Report
(where 1 is for Housekeeping and 2 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 1 and 2, is:
**SID** is the structure identification and by converting the integer to a string, it is possible to represent this field in M&C as:

```
AggregationDefinition.name == int2strAgg(SID)
```

**Collection Interval**, in M&C:

```
AggregationDefinition.updateInterval == Collection Interval * <DIAG_MIN_INTERV>
```

**NPAR1** is the number of parameters that are sampled once per Collection Interval, and in M&C it is not represented but not required because it is no longer a packet but an object.

**Parameter#** is the parameter number to be sampled once. In M&C it shall be represented by:

```
AggregationDefinition.parameterSets[1].parameters[n] == getPar(Parameter#[n])
```

(Where **n** represents the index of the array **Parameter#**)

**NFA** is the number of fixed-length arrays. In M&C, NFA+NPAR1 will be equal to the total number of elements in the **parameterSets** structure of the AggregationDefinition structure.

**NREP** is the number of values to be sampled for each parameter within this fixed-length array. In M&C, one shall use this parameter to calculate the sampleInterval field by dividing the Collection Interval by NREP:

```
AggregationDefinition.parameterSets[1+i].sampleInterval == Collection Interval / NREP[i] * <DIAG_MIN_INTERV>
```

(Where **i** represents the index for the fixed length arrays)

**NPAR2** is the number of different parameters within this fixed-length array, each of which shall be sampled NREP times per collection interval.

**Parameter##** is the parameter number to be sampled more than once (it was represented with two ‘#’ even though it appears with only one ‘#’ in the diagram). In M&C it shall be represented by:

```
AggregationDefinition.parameterSets[NPAR1+i].parameters[o] == getPar(Parameter##[i][o])
```

(Where **o** represents the index of each array **Parameter##[i]** within each fixed-length array)

When the request is received, the report definition is recorded and a corresponding “Report Generation Flag” is created. This flag is set to disable and the generation mode is “Periodic” by default. This means that in M&C the generationEnabled field shall be set to FALSE as default; the filterEnabled field shall be also set to FALSE, and the filterTimeout set to zero.

```
AggregationDefinition.generationEnabled = FALSE
AggregationDefinition.filterEnabled = FALSE
```

---

Page 13/78

Master Thesis
AggregationDefinition.filterTimeout = 0

### 3.2.3.1 PUS packet to M&C

For these two telecommands, the corresponding operation is **addDefinition**. This operation requires the input of a list of AggregationDefinition structure type. It is important to mention that this operation will return a list of MAL::Long which correspond to the object instance identifiers.

The representation of the AggregationDefinition structure to be used with the addDefinition operation:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td>int2strAgg (SID)</td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td>&quot;Operations 1 and 2&quot;</td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td>GENERAL/DIAGNOSTIC</td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td>FALSE</td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td>Collection Interval *</td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td>FALSE</td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td>0</td>
</tr>
<tr>
<td>parameterSets[1]</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters[n]</td>
<td>List<a href="">MAL::Long</a></td>
<td>getPar(Parameter#[n])</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td>o</td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>NULL</td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
<tr>
<td>parameterSets[NPAR1+i]</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters[o]</td>
<td>List<a href="">MAL::Long</a></td>
<td>getPar(Parameter##[i][o])</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td>Collection Interval/NREP[i] *</td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>NULL</td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
</tbody>
</table>

After generating the **aggregation** object, this can be used in the operation:

**addDefinition (aggregation)**

A general representation of the operations and inputs for this request follows:

<table>
<thead>
<tr>
<th>Operation</th>
<th>addDefinition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>List&lt; AggregationDefinition &gt;</td>
</tr>
</tbody>
</table>
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

<table>
<thead>
<tr>
<th>Input Data</th>
<th>aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>List<a href="">MAL::Long</a></td>
</tr>
</tbody>
</table>

It is important to mention that in PUS it is only possible to submit one parameter report definition per packet request while in M&C it is possible to submit a list of Aggregation definitions in a single operation.

### 3.2.3.1.2 M&C to PUS packet

The operation `addDefinition` can be mapped into a PUS packet telecommand by retrieving its input object data (type `AggregationDefinition`) and use its content for the generation of the packet:

It is necessary to count how many parameters exist with a `sampleInterval` field within the `parameterSets` set to 0 and insert this number respectively in the `NPAR1` packet field:

\[
\text{npar1} = \text{count_if}( \text{parametersSets}, \text{"parametersSets.sampleInterval == 0"} )
\]

It is required to insert these parameters with a `sampleInterval` equal to zero into an array in order to fill the `Parameter#` packet field.

It is required to count how many `parametersSets` exist with a `sampleInterval` field different than 0 and insert this number respectively in the `NFA` packet field:

\[
\text{nfa} = \text{count_if}( \text{parametersSets}, \text{"parametersSets.sampleInterval != 0"} )
\]

It is required to get this field by dividing the `updateInterval` by the `sampleInterval` from each different `parameterSets` with a `sampleInterval` different than zero:

\[
\text{nfa} = \text{updateInterval} / \text{parametersSets[i].sampleInterval}
\]

It is also required to count how many `parameters` per each `parameterSets` exist with a `sampleInterval` field different than 0 and insert these quantities respectively in each different `NPAR2` packet field:

\[
\text{npar2} = \text{count_if}( \text{parametersSets[i].parameters}, \text{"parametersSets.sampleInterval != 0"} )
\]

It is required to insert the parameters with a `sampleInterval` different than zero into an array of arrays in order to fill the `Parameter##[]` packet field.

In order to get the packet; the following pseudo-code could be used:

```plaintext
nfa = 0, npar1 = 0, npar2 = 0;
for (int i = 0; i < parametersSets.size(); i++) {
    if (parametersSets[i].sampleInterval == 0) {
        for (int j = 0; j < parametersSets[i].parameters.size(); j++) {
            parameterZeroId[npar1+j] = parametersSets[i].parameters[j];
            parameterZero[npar1+j] = retrieve (ParameterDefinition &
            parameterZeroId[npar1+j]);
            npar1++;
        }
    } else {
        for (int j = 0; j < parametersSets[i].parameters.size(); j++) {
            parameterNotZeroId[nfa][j] = parametersSets[i].parameters[j];
        }
    }
}
```
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

The computed variables from the previous code shall be inserted into the PUS packet:

<table>
<thead>
<tr>
<th>SID</th>
<th>Collection Interval</th>
<th>NPAR1</th>
<th>Parameter#</th>
</tr>
</thead>
<tbody>
<tr>
<td>npar1</td>
<td>aggregation.updateInterval / &lt;DIAG_MIN_INTERV&gt;</td>
<td></td>
<td>str2intAgg(\text{aggregation.name})</td>
</tr>
<tr>
<td>npar2</td>
<td></td>
<td></td>
<td>str2intAgg(\text{parameterZero[].name})</td>
</tr>
</tbody>
</table>

The category field in the AggregationDefinition structure will determine the service number to be used, 1 or 2. A “HOUSEKEEPING” value requires PUS request number 1 and a “GENERAL” requires PUS request number 2:

- aggregation[].category == HOUSEKEEPING => PUS Request number 1
- aggregation[].category == GENERAL => PUS Request number 2

3.2.3.2 \([3,3]\) and \([3,4]\) - (TC) Clearing Housekeeping/Diagnostic Report Definitions
(where 3 is for Housekeeping and 4 is for Diagnostic)

In PUS, the Telecommand packet for the requests 3 and 4, is:

```
<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

NSID is the number of SIDs that follow in the packet.

SID is the structure identification of the Definitions to be cleared.

3.2.3.2.1 PUS packet to M&C

For these two telecommands, the corresponding aggregation service operation used should be: removeDefinition. This operation requires the input of a list of MAL::Long type each one corresponding to the Identifier of the definition to be removed. These object instance identifiers shall be obtained by using the listDefinition operation:

```
removeDefinition ( listDefinition ( int2strAgg ( List<SID[1] ... SID[NSID]> ) ) )
```

A general representation of the operations and inputs for this request follows:
### 3.2.3.2.2 M&C to PUS packet

The operation `removeDefinition` shall be mapped into a PUS packet telecommand from its input data, type `List<MAL::Long>`, named `listData` on the following lines and use its content for the generation of the packet.

First, it is necessary to count how many elements are in the input list in order to fill the NSID packet field:

```
nsid = count (listData)
```

One has to use the operation `retrieve` from the COM services archive to get the Aggregation definitions for the `listData` list and then one shall obtain the required SID values by accessing the name field of each aggregation definition:

```
aggregation[] = retrieve (AggregationDefinition & listData)
```

From the previous variables, the PUS packet shall be generated:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>nsid</td>
<td><code>str2intAgg (aggregation[].name)</code></td>
</tr>
</tbody>
</table>

Once again, the service number to be used will be determined by the category field of each aggregation object:

- `aggregation[].category == HOUSEKEEPING` => PUS Request number 3
- `aggregation[].category == GENERAL` => PUS Request number 4

### 3.2.3.3 [3,5], [3,6], [3,7] and [3,8] - (TC) Enable/Disable Housekeeping/Diagnostic Parameter Report Generation

(Where 5 and 6 are for Housekeeping and where 6 and 7 are for Diagnostic) (5 and 7 enables and where 6 and 8 disables the Parameter Report Generation)

In PUS, the Telecommand packet, application data for the requests 5, 6, 7 and 8, is:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>nsid</td>
<td><code>Optional</code></td>
</tr>
</tbody>
</table>

**NSID** is the number of SIDs that follow in the packet.

**SID** is the structure identification of the Definitions to be enabled/disabled.
3.2.3.3.1 **PUS packet to M&C**

For these two telecommands, the corresponding aggregation service operation used should be: `enableGeneration`. This operation requires the input of a MAL::Boolean which indicates whether the supplied object instance identifiers (second input argument) are group objects or parameter objects. If `TRUE`, then the object instance identifier are group identifiers, else, shall be Parameter Definition identifiers. In PUS the concept of groups does not exist, which means that the first input MAL::Boolean shall be always set to `FALSE`:

```
enableGeneration ( FALSE & booleanPair )
```

<table>
<thead>
<tr>
<th>Operation</th>
<th>listDefinition</th>
<th>enableGeneration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>MAL::Boolean &amp; List<a href="">COM::InstanceBooleanPair</a></td>
</tr>
<tr>
<td>Input Data</td>
<td>int2strAgg ( SID[1] )</td>
<td>FALSE</td>
</tr>
<tr>
<td></td>
<td>int2strAgg ( SID[2] )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>int2strAgg ( SID[NSID] )</td>
<td></td>
</tr>
<tr>
<td>Output type</td>
<td>List<a href="">MAL::Long</a></td>
<td></td>
</tr>
</tbody>
</table>

Where, the \( n \)th element of the `booleanPair` list is:

<table>
<thead>
<tr>
<th>InstanceBooleanPair <code>booleanPair[n]</code></th>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>id</td>
<td>MAL::Long</td>
<td>listDefinition( int2strAgg ( SID[n] ) )</td>
</tr>
<tr>
<td></td>
<td>value</td>
<td>MAL::Boolean</td>
<td>TRUE/FALSE</td>
</tr>
</tbody>
</table>

It is important to mention that a value of `TRUE` in the value field of the InstanceBooleanPair object implies that updates of the parameters shall be generated, else, they will not be generated.

3.2.3.3.2 **M&C to PUS packet**

The operation `enableGeneration` shall be completely mapped into a PUS packet telecommand by analyzing its input data, type MAL::Boolean and List<COM::InstanceBooleanPair>. It was mentioned before that the first parameter indicates whether the supplied object instance identifiers are group objects or parameter objects. The group objects concept doesn’t exist in PUS, so, if the first argument of the operation is `TRUE`, then it will be necessary to get all the object instance identifiers from the group. If `FALSE`, then the list of InstanceBooleanPair will contain directly the object instance identifiers. On the following lines, it will be assumed that the operation `enableGeneration` is being executed with the arguments `group` (with type MAL::Boolean) and `booleanPair` (type List<COM::InstanceBooleanPair>):

```
enableGeneration ( group & booleanPair )
```

It is also necessary to check if the parameter to be send is an enable or disable instruction.

If the argument `group`, is `FALSE`, then:
- One has to use the operation `retrieve` from the COM services archive to get the Aggregation definitions for each id of the `booleanPair` list and then one shall obtain the required SID value by accessing the name field of each aggregation definition:

```
aggregation[] = retrieve ( AggregationDefinition & booleanPair[].id )
```
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

- It is necessary to divide the previously generated `aggregation[]` array in four different collections based on their category field (if HOUSEKEEPING then the requests are 5 or 6, else 7 or 8) and on their `booleanPair[].value` (if TRUE then the requests are 5 and 7, else 6 and 8).

  a. It is required to count how many elements there are in each collection, in order to fill the NSID packet field:

  \[
  \text{nsid} = \text{count}\left( \text{booleanPair} \right) 
  \]

  If the argument `group`, is `TRUE`, then:

  - The procedures will be essentially the same with the difference that it is required to access each aggregation from each element of the list of groups. These lists of aggregations will be in each one of the `booleanPair[].id.instanceIds` object array.

Reminder: it was assumed that the operation and its respective arguments are:

`enableGeneration (group & booleanPair)`

The code to generate the `objId` and `enabled` list of objects is available on the complete Systematic Comparison document.

\[
\text{aggregation} = \text{retrieve}\left( \text{AggregationDefinition} & \text{objId} \right);
\]

```java
for (int i=0; i < aggregation.size(); i++){
    if ( aggregation[i].category == \text{HOUSEKEEPING} && enabled[i] == \text{TRUE} ){
        sid5[] = add ( str2intAgg ( aggregation[i].name ) );
    }
    if ( aggregation[i].category == \text{HOUSEKEEPING} && enabled[i] == \text{FALSE} ){
        sid6[] = add ( str2intAgg ( aggregation[i].name ) );
    }
    if ( aggregation[i].category == \text{DIAGNOSTIC} && enabled[i] == \text{TRUE} ){
        sid7[] = add ( str2intAgg ( aggregation[i].name ) );
    }
    if ( aggregation[i].category == \text{DIAGNOSTIC} && enabled[i] == \text{FALSE} ){
        sid8[] = add ( str2intAgg ( aggregation[i].name ) );
    }
}
```

Finally, it is now possible to generate the four PUS packets:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>count(sidX[])</td>
<td>sidX[]</td>
</tr>
</tbody>
</table>

Where `sidX[]` can take one or more of the following arrays:

- `sidX[] = sid5[]` => PUS Request number 5
- `sidX[] = sid6[]` => PUS Request number 6
- `sidX[] = sid7[]` => PUS Request number 7
- `sidX[] = sid8[]` => PUS Request number 8
(where 9 is for Housekeeping and 11 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 9 and 11, is:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsigned Integer</td>
<td>Enumerated</td>
</tr>
</tbody>
</table>

NSID is the number of SIDs that follow in the packet.

SID is the structure identification of the Definitions to be Reported.

When this request is received, there is a generation of a Report containing the parameter definitions of all the SIDs demanded. These reports will be seen in more detail further on (services 10. and 12.).

3.2.3.4.1 PUS packet to M&C

For these two telecommands, there are no direct aggregation service operations, but it is possible to recreate the same Requests/Reports (9., 11., 10., and 12) by doing a request to the archive from the COM generic service. In order to do this, one starts by using M&C service operation: listDefinition. This will retrieve a list of MAL::Long which correspond to the object instance identifier definitions. And then, use this same list to query the COM archive using operation retrieve from COM:

\[ \text{retrieve (AggregationDefinition & listDefinition (int2strAgg (List<& SID[1] ... SID[NSID]>)))} \]

<table>
<thead>
<tr>
<th>Operation</th>
<th>Input type</th>
<th>Input Data</th>
<th>Output type</th>
</tr>
</thead>
<tbody>
<tr>
<td>listDefinition</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>int2strAgg (SID[1])</td>
<td>List<a href="">MAL::Long</a></td>
</tr>
<tr>
<td>retrieve</td>
<td>ObjectType &amp; List<a href="">MAL::Long</a></td>
<td>AggregationDefinition int2strAgg (SID[2])</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>int2strAgg (SID[NSID])</td>
<td></td>
</tr>
</tbody>
</table>

3.2.3.4.2 M&C to PUS packet

In M&C, the Aggregations are stored by their name identifiers (previously considered to be a string of digits in order to allow the conversion). It is necessary to convert the name field of the aggregation definition and then insert it into the PUS packet SID field. Regarding the PUS packet field NSID, this shall be filled by counting the number of SIDs used. This operation in M&C is useless because the aggregations are stored in the Archive of the COM services, so there is no need to do this.

Although the mapping of this operation will never occur, the corresponding PUS packet would be:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
</tr>
</thead>
<tbody>
<tr>
<td>count (aggregationDefinition)</td>
<td>str2intAgg (List&lt;aggregationDefinition.name&gt;)</td>
</tr>
</tbody>
</table>
### 3.2.3.5 [3,10] and [3,12] - (TM) Housekeeping/Diagnostic Parameter Report Definitions Report

(where 10 is for Housekeeping and 12 is for Diagnostic)

In PUS, the Telemetry source packet, source data for the reports 10 and 12, is:

<table>
<thead>
<tr>
<th>NSID</th>
<th>SID</th>
<th>Collection Interval</th>
<th>NPAR1</th>
<th>Parameter#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsigned Integer</td>
<td>Enumerated</td>
<td>Unsigned Integer</td>
<td>Unsigned Integer</td>
<td>Enumerated</td>
</tr>
</tbody>
</table>

- **NSID** is the number of SIDs that follow in the packet.
- **Collection Interval**, it is the equivalent to the updateInterval field.
- **NPAR1** is the number of parameters that are sampled once per Collection Interval, and in M&C it is not represented but not required because it is no longer a packet but an object.
- **Parameter#** is the parameter number to be sampled once.

<table>
<thead>
<tr>
<th>NFA</th>
<th>NREP</th>
<th>NPAR2</th>
<th>Parameter#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsigned Integer</td>
<td>Unsigned Integer</td>
<td>Unsigned Integer</td>
<td>Enumerated</td>
</tr>
</tbody>
</table>

- **NFA** is the number of fixed-length arrays. This concept is not used and not required in M&C because the parameterSets is a list of MAL::AggregationReference field type.
- **NREP** is the number of values to be sampled for each parameter within this fixed-length array. In M&C, one shall use this parameter to calculate the sampleInterval field by dividing the Collection Interval by NREP.
- **NPAR2** is the number of different parameters within this fixed-length array, each of which shall be sampled NREP times per collection interval.
- **Parameter##** is the parameter number to be sampled more than once (it was represented with two ‘#’ even though it appears with only one ‘#’).

#### 3.2.3.5.1 PUS packet to M&C

For these telemetry packets, it is possible to insert its data into an AggregationDefinition structure from M&C. These telemetry packets are only generated after sub-services 9 and 11 requests them. In the previous section, 3.2.3.4.1, it was shown that the operation retrieve from the COM book combined with the listDefinition operation, would output a list of Aggregation definitions.
For one PUS telemetry packet, the corresponding M&C AggregationDefinition structure would be:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td>int2strAgg (SID[1])</td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td>&quot;Report for PUS 10 and 12&quot;</td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td>GENERAL/DIAGNOSTIC</td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td>TRUE</td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td>Collection Interval * &lt;DIAG_MIN_INTERV&gt;</td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td>FALSE</td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td>0</td>
</tr>
<tr>
<td>parameterSets[n]</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters</td>
<td>List<a href="">MAL::Long</a></td>
<td>getPar(Parameter#[n])</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td>0</td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>NULL</td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
<tr>
<td>parameterSets[NPAR1+i]</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters[i][o]</td>
<td>List<a href="">MAL::Long</a></td>
<td>getPar(Parameter##[i][o])</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td>Collection Interval / NREP[i] *&lt;DIAG_MIN_INTERV&gt;</td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>NULL</td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
</tbody>
</table>

### 3.2.3.5.2 M&C to PUS packet

The structure of this PUS packet is the same as in section 3.2.3.1, with the only difference being at the beginning of the packet with one more field: NSID. This field allows to define how many repetitions of the following field will exist. This means that it is possible to report multiple different definitions instead of only one.

To avoid the repetition of text, please check section 3.2.3.1. The packet in that section will be repeated NSID times. So, the code will be essentially the same with the difference that it has to be looped NSID times.
3.2.3.6 [3,13] and [3,14] - (TC) Report Housekeeping/Diagnostic Parameter Sampling-Time Offsets
(where 13 is for Housekeeping and 14 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 13 and 14, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Enumerated</th>
</tr>
</thead>
</table>

For this service there is no direct equivalent M&C operation because all the AggregationValue objects are stamped with this time in the intervalTime field, but it is possible to get only the Sampling-Time Offsets by performing a series of operations. These will be described on the next sections.

3.2.3.6.1 PUS packet to M&C

This request doesn’t exist in M&C so there shall be no M&C representation. These service requests (13 and 14) are intrinsically connected with the next two reports (15 and 16) so the reader should check the section 3.2.3.7, for a complete understanding of how to get the equivalent Sampling-Time Offset on a request.

3.2.3.6.2 M&C to PUS packet

To generate this packet, it is only required to do a conversion from the string name of the Aggregation definition to an integer:

| SID | str2intAgg (aggregation.name) |

3.2.3.7 [3,15] and [3,16] - (TM) Housekeeping/Diagnostic Parameter Sampling-Time Offsets Report
(where 15 is for Housekeeping and 16 is for Diagnostic)

In PUS, the Telemetry source packet, source data for the reports 15 and 16, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Time Offset 1st Parameter</th>
<th>...</th>
<th>Time Offset Last Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enumerated</td>
<td>Relative Time</td>
<td>...</td>
<td>Relative Time</td>
</tr>
</tbody>
</table>

The (NPAR1 + NPAR2) time offsets correspond to the order of the parameters in the corresponding housekeeping or diagnostic parameter report. For a fixed-length array, only the time offset of the first occurrence of a parameter within the array shall be reported.

3.2.3.7.1 PUS packet to M&C

In M&C, there are no direct aggregation service operations for this packet, because all the AggregationValue objects are stamped with this time in the intervalTime field. The closest M&C operation is getValue, which returns the latest value for a requested aggregation.
If one needs to execute such telecommand, one shall do a sequence of operations in M&C to obtain the same result. It will consist in using the aggregation service operation `getValue`, in order to retrieve an AggregationValue object where one shall access its intervalTime field. For this, like in the previous requests studied in the previous sections, one needs to use the `listDefinition` operation once again. The sequence of operations would be:

```
returnedValue = getValue ( listDefinition ( int2strAgg (SID) ) )
```

Where the output, `returnedValue`, would be a list of objects type AggregationValue. The intervalTime field of this object, corresponds to the Time Offset field in PUS:

```
returnedValue[n].intervalTime == (Time Offset)[n]
```

### 3.2.3.7.2 M&C to PUS packet

To generate this packet, one shall use the M&C sequence of operations in the previous section and obtain once again the list `returnedValue` with type AggregationValue and fill in the PUS packet the following way:

<table>
<thead>
<tr>
<th>SID</th>
<th>Time Offset 1st Parameter</th>
<th>…</th>
<th>Time Offset Last Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>str2intAgg</td>
<td>returnedValue[1].intervalTime</td>
<td>…</td>
<td>returnedValue[n].intervalTime</td>
</tr>
<tr>
<td>(aggregation.name)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is important to mention that the order of the time offsets might not be correct. The first parameters should have a sampleInterval equal to the updateInterval of the aggregation which would correspond to the first parameters in the PUS packet with a NREP equal to zero (check 3.2.3.7).

### 3.2.3.8 [3,17] and [3,18] - (TC) Select Periodic Housekeeping/Diagnostic Report Generation Mode

(where 17 is for Housekeeping and 18 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 17 and 18, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Enumerated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SID</td>
</tr>
</tbody>
</table>

When this request is received, the parameter report generation mode for the indicated reporting definition shall be changed to periodic mode. If the reporting definition is already in periodic mode, an error report shall be generated.

### 3.2.3.8.1 PUS packet to M&C

For these two telecommands, the corresponding aggregation service operation used should be: `enableFilter`. This operation requires the input of a MAL::Boolean which indicates whether the supplied object instance identifiers are group objects or parameter objects. If TRUE, then the object instance identifier are group identifiers, else, shall be Aggregation Definition identifiers. In PUS the concept of groups does not exist, which means that the first input MAL::Boolean shall be always set to FALSE:

```
enableFilter ( FALSE & InstanceBooleanPair (listDefinition (int2strAgg (SID)), FALSE) )
```

<table>
<thead>
<tr>
<th>Operation</th>
<th>listDefinition</th>
<th>enableFilter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>MAL::Boolean &amp; List<a href="">COM::InstanceBooleanPair</a></td>
</tr>
</tbody>
</table>
### 3.2.3.8.2 M&C to PUS packet

As mentioned in 3.2.3.8.1, the equivalent M&C operation shall be `enableFilter`, which takes two parameters as input, a MAL::Boolean and a list of COM::InstanceBooleanPair. The first parameter defines if the id of the InstanceBooleanPair, second argument of the `enableFilter` operation are group objects or if they are Aggregation Definition identifiers.

PUS only allows the commutation of one Aggregation Definition per telecommand. This implies that if the second argument of the `enableFilter` contains in the id field a list of aggregation definitions or if it contains groups, then it will be necessary to execute the telecommand for each one of the aggregations in the list or for each one of the aggregations in each group.

Let’s consider that the operation `enableFilter` was executed with the following arguments:

```c
enableFilter ( group & boolPair )
```

The code to generate the `objId` and `enabled` list of objects is available on the complete Systematic Comparison document.

```c
aggregation = retrieve (AggregationDefinition & objId);
```

The PUS packet has to be sent \( N \) times for all the different aggregations objects in the list where the corresponding `enabled[N]` object is set to `FALSE`:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>MAL::Long</td>
<td><code>listDefinition (int2strAgg (SID))</code></td>
</tr>
<tr>
<td>value</td>
<td>MAL::Boolean</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

A value field set to FALSE in the InstanceBooleanPair object means that the updates will not be filtered, which is the equivalent of selecting the Periodic mode in PUS.

The category field in the AggregationDefinition structure will determine the service number to be used, 1 or 2. A “HOUSEKEEPING” value requires PUS request number 17 and a “GENERAL” requires PUS request number 18:

```c
aggregation[N].category == HOUSEKEEPING => PUS Request number 17
aggregation[N].category == GENERAL => PUS Request number 18
```

The previous telecommand can only be sent if the corresponding `enabled[N]` object is set to `FALSE`. If it is set to `TRUE`, then it means that the `enableFilter` operation is enabling the filter and so, to do the same in PUS, one should execute the telecommand from the next section, 3.2.3.9.
3.2.3.9 [3,19] and [3,20] - (TC) Select Filtered Housekeeping/Diagnostic Report Generation Mode
(where 19 is for Housekeeping and 20 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 19 and 20, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Timeout</th>
<th>N</th>
<th>Parameter#</th>
<th>Threshold Type</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enumerated</td>
<td>Unsigned Integer</td>
<td>Unsigned</td>
<td>Enumerated</td>
<td>Enumerated</td>
<td>Unsigned Integer</td>
</tr>
</tbody>
</table>

When this request is received, the parameter report generation mode for the indicated reporting definition shall be changed to Filtered mode.

**Timeout** is the timeout of a one-shot parameter report expressed as a multiple of the data collection interval for this reporting definition. This means that in M&C, its corresponding value shall be:

\[\text{AggregationDefinition.filteredTimeout} = \text{Threshold} \times \text{AggregationDefinition.updateInterval}\]

**N** is the number of parameters and their threshold attributes which follow. In M&C will be the number of parameters in every parameterSets combined.

**Parameter#** are the parameters which are unfiltered when the values of samples between successive collection intervals are compared.

**Threshold Type** indicates the type of the threshold value that follows. This shall be either 0, for a percentage or 1, for an absolute delta value. In M&C, this shall be represented by selecting in the thresholdType field of the periodicFilter in the parameterSets which belongs to the AggregationDefinition structure, the value of PERCENTAGE or DELTA.

\[\text{AggregationDefinition.parameterSets[\#].periodicFilter.thresholdType} = \text{PERCENTAGE}\]
\[\text{AggregationDefinition.parameterSets[\#].periodicFilter.thresholdType} = \text{DELTA}\]

**Threshold** is the minimum change to be detected between parameter samples in order to declare an event for the purpose of parameters report generation. In M&C, this shall be represented by selecting in the thresholdValue field of the periodicFilter in the parameterSets which belongs to the AggregationDefinition structure, the value from the PUS.

\[\text{AggregationDefinition.parameterSets[\#].periodicFilter.thresholdValue} = \text{Threshold}\]

“Select Periodic Report Generation mode” in PUS, destroys the configurations for the filtered mode, which means that if one wants to activate the filtered mode, one has to send all the configuration parameters for the filter once again, just like it was shown above. In M&C it is allowed to enable/disable the periodic mode without destroying the filter configurations with the function \text{enableFilter}, but this shall not be used. Instead it is necessary to use the function \text{updateDefinition}, because it allows the change to filtered mode directly with the parameters within the PUS telecommand request.

### 3.2.3.9.1 PUS packet to M&C

For these two telecommands, there are no direct aggregation service operations, but it is possible to recreate the same Requests (19 and 20) by querying the archive from the COM generic service to get the AggregationDefinition and then updating the definition with operation, \text{updateDefinition}. In order to do this, one starts by using M&C service: \text{listDefinition}. This will retrieve a list of MAL::Long which
correspond to the object instance identifier definitions. And then, use this same list to query the COM archive using operation `retrieve` from COM:

```
definition = retrieve ( AggregationDefinition & listDefinition (int2strAgg (SID)) )
```

After this, one shall update the `definition` object according to the PUS packet fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td>int2strAgg (SID)</td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td>&quot;Definition for 19 and 20&quot;</td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td></td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td>TRUE</td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td>Timeout * updateInterval</td>
</tr>
<tr>
<td>parameterSets[n]</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters</td>
<td>List<a href="">MAL::Long</a></td>
<td>getPar(Parameter#[n])</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>Threshold Type[n]</td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td>Threshold[n]</td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
</tbody>
</table>

(With N elements n)

Empty fields shall have the same values as the unmodified definition object.

Now, it is necessary to use the aggregation service operation, `updateDefinition`, in order to send the updated `definition` object:

```
updateDefinition ( listDefinition (int2strAgg (SID)) & definition )
```

### 3.2.3.9.2 M&C to PUS packet

To enable filtered reports in M&C it can be used both the operation `enableFilter`, and/or `updateDefinition`. The first one will just change the filteredEnabled field in the aggregation definition stored while the second one allows a bigger set of changes in the aggregation definition.

If the operation `enableFilter` is used with the boolPair[].value field set to TRUE and group set to FALSE:

```
enableFilter ( group & boolPair )
```

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>MAL::Long</td>
<td>...</td>
</tr>
<tr>
<td>value</td>
<td>MAL::Boolean</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

Then, the definitions to be used in the PUS packet should be retrieved from the archive:

```
definition = retrieve ( AggregationDefinition & listDefinition (boolPair[].id) )
```
The PUS packet [3.19] or [3.20] (Housekeeping or Diagnostic) shall then be generated \( N \) times:

<table>
<thead>
<tr>
<th>SID</th>
<th>Timeout</th>
<th>( N )</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>str2intAgg</code> (definition[n].name)</td>
<td><code>definition[n].filteredTimeout / \( \text{filteredTimeout} \)</code></td>
<td>( \text{count} ) (definition[n].parameterSets)</td>
</tr>
</tbody>
</table>

If the flag `group` is set to `FALSE`, then the previous PUS packet shall be recreated for each one of the aggregations in each one of the groups, just like occurred in section 3.2.3.8.2.

For the second operation, `updateDefinition`, let’s consider the following arguments:

\[ \text{updateDefinition} \left( \text{sid} \& \text{newDefinition} \right) \]

According to the M&C services book, the operation `updateDefinition`, allows the user to modify one or more aggregations without needing to delete the existing definition first. It is only expected to be used for minor modifications.

This M&C operation has no direct equivalent to PUS but it can be fully mapped into PUS by combining a sequence of PUS requests if needed. For this, one starts by comparing the old definition (`oldDefinition`) with the new one (`newDefinition`). From the archive:

\[ \text{oldDefinition} = \text{retrieve} \left( \text{AggregationDefinition} \& \text{sid} \right) \]

If the change is in the field description, nothing shall be done. If the change occurs in the field category or `updateInterval`, then it is necessary to delete the definition and add it again, because PUS doesn’t allow the modification of these parameters. To do this, the requests 3 and 4 for should be used to delete the definition and the requests 1 and 2 to add the new one (similar to the previous but with some modifications).

If there is a change in the `generationEnabled` field, then, one should verify if the change was to `TRUE` or to `FALSE`. If `FALSE`, telecommands 6 or 8 shall be sent, if `TRUE`, telecommands 5 or 7 shall be sent (depends if it is `HOUSEKEEPING` or `DIAGNOSTIC`).

Now depending on the `filteredEnabled` field of `newDefinition`, if `FALSE`, nothing shall be done, else the request 19 and 20 will be sent (check first part of this section).

### 3.2.3.10 [3,21] and [3,22] - (TC) Report Unfiltered Housekeeping/Diagnostic Parameters

(where 21 is for Housekeeping and 22 is for Diagnostic)

In PUS, the Telecommand packet, application data for the requests 21 and 22, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Enumerated</th>
</tr>
</thead>
</table>

When this request is received, a report shall be generated as the one on the next section 3.2.3.11.
These service requests (21 and 22) are intrinsically connected with the next two reports, 23 and 24, so the reader should check the next section for a complete understanding of how to get the Report of parameters on request.

3.2.3.10 **M&C to PUS packet**

To generate the equivalent PUS packet from the M&C, one starts by retrieving from the archive:

\[\text{definition} = \text{retrieve} \left( \text{AggregationDefinition} & \text{sid} \right)\]

Then, it is possible to generate the equivalent PUS request packet:

\[
\begin{array}{|c|c|}
\hline
\text{SID} & \text{str2intAgg} \\
\text{definition.name} & \\
\hline
\end{array}
\]

3.2.3.11 **[3,23] and [3,24] - (TM) Unfiltered Housekeeping/Diagnostic Parameter Report**

(where 23 is for Housekeeping and 24 is for Diagnostic)

In PUS, the Telemetry source packet, source data for the reports 23 and 24, is:

\[
\begin{array}{|c|c|c|c|}
\hline
\text{SID} & \text{N} & \text{Parameter#} & \text{Threshold} \\
\text{Enumerated} & \text{Unsigned Integer} & \text{Enumerated} & \text{Enumerated} & \text{Unsigned integer} \\
\hline
\end{array}
\]

The parameters description for this packet were omitted because they were already mentioned in the previous section 3.2.3.9.

3.2.3.11.1 **PUS packet to M&C**

In M&C, one shall do a sequence of operations which allow the user to obtain the same result. It will consist in using the COM operation `retrieve`, in order to retrieve an AggregationDefinition object which resembles the telemetry packet 23 and 24. Like the requests studied in the previous sections, the `listDefinition` operation combined with the `retrieve` operation from the COM services are used once again. The sequence of operations would be:

\[\text{definition} = \text{retrieve} \left( \text{AggregationDefinition} & \text{listDefinition \left( \text{int2strAgg \left( \text{SID} \right) } \right)} \right)\]

Where the output, `definition`, shall be a list with a single element type `AggregationDefinition`.

It is expected that the `definition` object shall be:

\[
\begin{array}{|c|c|c|}
\hline
\text{Field} & \text{Type} & \text{Value} \\
\text{name} & \text{MAL::Identifier} & \text{int2strAgg \left( \text{SID} \right)} \\
\hline
\end{array}
\]
3.2.3.11.2 **M&C to PUS packet**

In M&C, the definitions are all stored in the archive, so:

\[
\text{definition} = \text{retrieve}(\text{AggregationDefinition} \& \text{listDefinition}(\text{int2strAgg}(\text{SID})))
\]

Now, the PUS packet shall be generated the following way:

<table>
<thead>
<tr>
<th>SID</th>
<th>N</th>
<th>Parameter#</th>
</tr>
</thead>
<tbody>
<tr>
<td>str2intAgg (name)</td>
<td>count (parameterSets)</td>
<td>str2intAgg (parameterSets[].parameters[].name)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Threshold Type</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterSets[].periodicFilter.thresholdType</td>
<td>parameterSets[].periodicFilter.thresholdValue</td>
</tr>
</tbody>
</table>

With all the previous variables coming from the definition object obtained previously.

3.2.3.12 **[3,25] and [3,26] - (TM) Housekeeping/Diagnostic Parameter Report**

(where 25 is for Housekeeping and 26 is for Diagnostic)

In PUS, the Telemetry source packet, source data for the reports 25 and 26, is:

<table>
<thead>
<tr>
<th>SID</th>
<th>Mode</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enumerated</td>
<td>Enumerated</td>
<td>Any</td>
</tr>
</tbody>
</table>

**Mode**, it indicates the mode of packet generation and, if filtered, the reason for generation of this packet.

- **Mode** == 0: Periodic mode;
- **Mode** == 1: Filtered mode, parameter change exceeded threshold;
- **Mode** == 2: Filtered mode, but packet generated as a result of timeout.

In M&C, their respective equivalence is:
Mode == 0:

AggregationValue.generationMode = PERIODIC
AggregationValue.filtered = FALSE

Mode == 1:

AggregationValue.generationMode = PERIODIC
AggregationValue.filtered = TRUE

Mode == 2:

AggregationValue.generationMode = FILTERED_TIMEOUT
AggregationValue.filtered = TRUE

Parameters, this field consists of a sequence of values of parameters that are sampled once per collection interval followed by a sequence of fixed-length arrays of records. The sequence of parameter values and arrays shall be the same as in the housekeeping or diagnostic parameter report definition request.

3.2.3.12.1 PUS packet to M&C

This type of packets will be equivalent to the generated objects by the M&C operation service, monitorValue, when it outputs an AggregationValue object type.

One starts by retrieving from the archive the corresponding SID:

\[
\text{definition} = \text{retrieve} (\text{AggregationDefinition} \& \text{listDefinition (int2strAgg (SID))})
\]

The AggregationValue.deltaTime field can be set to NULL and that would represent that the sample time is the same as the aggregation time stamp. This field must be filled by taking advantage of the PUS requests [3,13] and [3,14] which will report the Time-offset intervals (check section 3.2.3.6.1 for the filling).

In M&C, Filtering is where an aggregation is only generated when the change in the value of the filtered parameters exceeds a specified threshold or a timeout is passed. This directly implies that when the filtering is on, there will be only one sampling of the parameters.

If AggregationValue.filtered == TRUE, then:

AggregationValue.intervalTime = NULL
AggregationValue.setIntervalTime = NULL

If AggregationValue.filtered == FALSE, then:

AggregationValue.intervalTime = definition.parameterSets.sampleInterval
AggregationValue.setIntervalTime = definition.parameterSets.sampleInterval \times \text{count (AggregationValue.Values)}

PUS doesn’t have the concept of validity expressions, which means that the equivalent mapping of the validityState field should be always set to VALID:

AggregationValue.Values.validityState = VALID

In PUS, there’s no conversion of values which means that the parameters are always in the raw format:

AggregationValue.rawValue = Parameters[n]

PUS doesn’t have the concept of conversions, so the equivalent mapping of the convertedValue field shall be always set to NULL:

AggregationValue.convertedValue = NULL

According to the previous conditions, the AggregationValue structure shall then be:
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>generationMode</td>
<td>GenerationMode</td>
<td>PERIODIC/FILTERED_TIMEOUT</td>
</tr>
<tr>
<td>filtered</td>
<td>MAL::Boolean</td>
<td>TRUE/FALSE</td>
</tr>
<tr>
<td>deltaTime</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>intervalTime</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>setIntervalTime</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>values[n]</td>
<td>List&lt;ParameterValue&gt;</td>
<td></td>
</tr>
<tr>
<td>validityState</td>
<td>Validity</td>
<td>VALID</td>
</tr>
<tr>
<td>rawValue</td>
<td>MAL::Attribute</td>
<td>Parameters[n]</td>
</tr>
<tr>
<td>convertedValue</td>
<td>MAL::Attribute</td>
<td>NULL</td>
</tr>
</tbody>
</table>

(Where \( n \) represents the element number of the Parameters field in the PUS packet)

The related field of the AggregationValue should be pointing to the definition which generated it:

\[
\text{AggregationValue.related} = \text{listDefinition (int2strAgg (SID))}
\]

### 3.2.3.12.2 M&C to PUS packet

In M&C, the operation `monitorValue`, returns two parameters, the first has a COM::ObjectId type and the second has an AggregationValue type. To map the output of this function into a PUS packet, one starts be getting the SID of the packet from the first parameter. Let’s assume that:

\[
\text{objectId} \& \text{aggregationValue} = \text{monitorValue()}
\]

From the `objectId` it is possible to obtain the aggregation definition which generated the aggregationValue:

\[
\text{definition} = \text{retrieve (AggregationDefinition \& objectId.id )}
\]

If the AggregationValue.filtered field is `FALSE`, then:

\[
\text{mode} = 0
\]

If the filtered field of the object `aggregationValue` is `TRUE`, then it is necessary to check the generationMode field in the `aggregationValue` object:

- If the AggregationValue.generationMode field is `PERIODIC`, then:
  \[
  \text{mode} = 1
  \]
- If the AggregationValue.generationMode field is `FILTERED_TIMEOUT`, then:
  \[
  \text{mode} = 2
  \]

It is now possible to generate the PUS packet:

<table>
<thead>
<tr>
<th>SID</th>
<th>Mode</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>str2intAgg (definition.name)</td>
<td>mode</td>
<td>definition.values[n].rawValue</td>
</tr>
</tbody>
</table>
3.2.4 **PUS-C: Requests and Reports**

In this section of the document it will be presented the PUS-C Requests and Reports which are not included in the PUS issue A. In service 3, the differences from PUS to PUS-C are the additional capability on the housekeeping and diagnostic subservices for generating one shot of housekeeping/diagnostic parameter reports; the capability of appending parameters to the housekeeping parameter report definitions; capability for modifying the housekeeping/diagnostic parameters report definitions; the capability for modifying the housekeeping parameter report collection intervals; and the capability for reporting the periodic generation action status of some housekeeping/diagnostic parameter report definitions.

There is also the presence of a new **Parameter reporting configuration subservice** which provides the capability to control the generation of the parameter reports generated by the housekeeping and the diagnostic reporting subservices. It operates on sets of parameter reports, of housekeeping or diagnostic nature; e.g. enabling or disabling the generation of such sets.

In MO M&C there are no direct equivalent for the Parameter reporting configuration subservice in the Aggregation service, so the Requests and Reports for cannot be mapped from or to MO M&C. It is not possible to use the group service to represent the Parameter reporting configuration subservice because it only allows the grouping of objects and nothing else. The group service neither stores the groups on the provider nor on the archive.

There is also a small change on the packets from [3,3] to [3,9], and [3,11], the NSID field is now named N field and it is no longer optional, it is mandatory. On packets [3,10] and [3,12], the NSID field no longer exists.

The PUS packets from [3,13] to [3,24] no longer exist in PUS-C, these were considered obsolete.

The PUS packet [3,25] and [3,26] also suffered some changes, the “Mode” field no longer exists.

A brief summary will still be presented for the PUS-C Requests and Reports which cannot be mapped.

A summary table for PUS-C follows:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>COM Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>(TC) Generate one shot Report for some Housekeeping Parameter Report Definitions</td>
<td>getValue()</td>
</tr>
<tr>
<td>28</td>
<td>(TC) Generate one shot Report for some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>(TC) Append Parameters to a Housekeeping Parameter Report Definition</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>30</td>
<td>(TC) Append Parameters to a Diagnostic Parameter Report Definition</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(TC) Modify the Collection Interval of some Housekeeping Parameter Report Definitions</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>32</td>
<td>(TC) Modify the Collection Interval of some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>(TC) Report the Periodic Generation Properties of some Housekeeping Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>(TC) Report the Periodic Generation Properties of some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>(TM) Housekeeping Parameter Report Definition Periodic Generation Properties Report</td>
<td>listDefinition()</td>
</tr>
<tr>
<td>36</td>
<td>(TM) Diagnostic Parameter Report Definition Periodic Generation Properties Report</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>(TC) Apply some Functional Parameter Reporting Configurations</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>(TC) Create a Functional Parameter Reporting Definition</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>(TC) Delete some Functional Parameter Reporting Definitions</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>(TC) Report some Functional Parameter Reporting Definitions</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>(TM) Functional Parameter Reporting Definition Report</td>
<td></td>
</tr>
</tbody>
</table>
3.2.4.1 [3,27] and [3,28] - (TC) Generate one shot Report for some Housekeeping/Diagnostic parameter Report Definitions (where 27 is for Housekeeping and 28 is for Diagnostic)

In PUS-C, the Telecommand packet, application data for the report 27 and 28, is:

\[
\text{repeated } N \text{ times}
\]

\[
\begin{array}{|c|c|}
\hline
N & \text{housekeeping/diagnostic parameter report definition ID} \\
\hline
\text{unsigned integer} & \text{enumerated} \\
\hline
\end{array}
\]

N is the number of different housekeeping parameter report definitions to be generated once.

\text{housekeeping/diagnostic parameter report definition ID (prdID), is the identifier of the housekeeping parameter report definition to be generated.}

3.2.4.1.1 \textit{PUS packet to M&C}

In M&C, the corresponding operation is \texttt{getValue()} which returns the latest value for a requested aggregation. The request includes a list of aggregation definition object instance identifiers:

\[
\text{returnedValue} = \text{getValue} \left( \text{listDefinition} \left( \text{int2strAgg(\{prdID[1] \ldots prdID[N]\})} \right) \right)
\]

Where the output, \texttt{returnedValue}, would be two lists. The first list contains the matched aggregation definition object instance identifiers and the second list contains the matched aggregation values.

A general representation of the operations and inputs for this request follows:

<table>
<thead>
<tr>
<th>Operation</th>
<th>listDefinition</th>
<th>getValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>List<a href="">MAL::Long</a></td>
</tr>
<tr>
<td>Input Data</td>
<td>int2strAgg(prdID[1])</td>
<td>PRD_ID[1]</td>
</tr>
<tr>
<td></td>
<td>int2strAgg(prdID[2])</td>
<td>PRD_ID[2]</td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td></td>
<td>int2strAgg(prdID[N])</td>
<td>PRD_ID[NSID]</td>
</tr>
<tr>
<td>Output type</td>
<td>List<a href="">MAL::Long</a></td>
<td>List<a href="">MAL::Long</a> &amp; List&lt;AggregationValue&gt;</td>
</tr>
</tbody>
</table>

(Both lists have a size N)
### 3.2.4.1.2 **M&C to PUS packet**

As mentioned before, the equivalent operation is `getValue()`. We shall assume that the operation was requested with the following parameter:

```
getValue ( definitionIds )
```

One shall retrieve the `definitionIds` from the archive:

```
aggregation = retrieve ( AggregationDefinition & definitionIds )
```

It is now possible to generate the PUS-C packet:

<table>
<thead>
<tr>
<th>N</th>
<th>parameter report definition ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>count (definitionsIds)</td>
<td>str2intAgg (aggregation[].name)</td>
</tr>
</tbody>
</table>

### 3.2.4.2 **[3,29] and [3,30] - (TC) Append Parameters to a Housekeeping/Diagnostic Parameter Report Definition**

(where 29 is for Housekeeping and 30 is for Diagnostic)

In PUS-C, the Telecommand packet, application data for the report 29, is:

```
repeated NFA times
```

```
repeated N1 times
```

```
repeated N2 times
```

<table>
<thead>
<tr>
<th>housekeeping/diagnostic parameter report definition ID</th>
<th>parameter ID</th>
<th>NFA</th>
<th>super commutated sample repetition number</th>
<th>N2</th>
<th>parameter ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated unsigned integer</td>
<td>enumerated unsigned integer</td>
<td>unsigned integer</td>
<td>unsigned integer</td>
<td>unsigned integer</td>
<td>enumerated</td>
</tr>
</tbody>
</table>

**N1** is the number of parameters that are sampled once per Collection Interval.

**parameter ID**, is the identifier of the parameter to be sampled once per collection interval.

**NFA** is the number of fixed-length arrays.

**Super commutated sample repetition number (NREP)** is the number of samples to be collected per Collection Interval.

**N2** is the number of different parameters within this fixed-length array, each of which shall be sampled “Super commutated sample repetition number” times per collection interval.
There’s no direct equivalent in M&C but it is possible to obtain the same result using the `updateDefinition()` operation. The operation is not exactly the same but it also allows the consumer to add new definitions to a specific aggregation like this PUS-C packet.

### 3.2.4.2.1 PUS packet to M&C

In M&C, the corresponding operation is `updateDefinition()` which allows a consumer to modify a definition for one or more aggregations without needing to delete the existing definition first.

To execute the equivalent PUS-C request, one needs to perform a series of operations. One shall start by getting the corresponding object instance identifier of the definition:

```plaintext
definitionId = listDefinition(int2strAgg(prdID))
```

After, one shall retrieve the respective aggregation from the archive:

```plaintext
aggregation = retrieve(AggregationDefinition & definitionId)
```

The new parameters shall be added to the parameterSets object. The next operation is necessary just for the understanding of the AggregationDefinition object presented on the next few lines:

```plaintext
nparam = count(aggregation.parameterSets)
```

The new parameters shall be appended on the `aggregation` definition object. Empty fields shall remain with the previous information on the original aggregation:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td></td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td></td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>parameterSets[nparam+n]</td>
<td>List<a href="">MAL::AggregationReference</a> &gt;</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>Parameter ID#[n]</td>
</tr>
<tr>
<td>parameters</td>
<td>List<a href="">MAL::Long</a></td>
<td>0</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td>NULL</td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td>ThresholdType</td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
<tr>
<td>parameterSets[nparam+N1+i]</td>
<td>List<a href="">MAL::AggregationReference</a> &gt;</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>Parameter ID#*[i][o]</td>
</tr>
<tr>
<td>parameters*[o]</td>
<td>List<a href="">MAL::Long</a></td>
<td>Collection Interval / NREP[i] * &lt;DIAG_MIN_INTERV&gt;</td>
</tr>
</tbody>
</table>
### 3.2.4.2.2 M&C to PUS packet

M&C does not have a dedicated operation to add parameter to a specific aggregation but it can be done using the `updateDefinition` operation. This means that for this mapping, one has to consider that the aggregation object in the `updateDefinition` operation is just adding new parameterSets to the definition without changing any of the other fields or previous parameters (similar to the `aggregation` object in section 3.2.4.2.1):

```plaintext
updateDefinition (aggregation)
```

One starts by checking the above conditions by comparing the modified `aggregation` object with the original object:

```plaintext
originalID = listDefinition (aggregation.name)
original = retrieve (AggregationDefinition & originalID)
```

One shall generate a list of new parameters added on the modified aggregation:

```plaintext
offset = count (original.parameterSets)
for (i: aggregation.parameterSets){
    listAdded[i] = aggregation.parameterSets[offset + i]
}
```

This new object can now be used in the code in section 3.2.3.1.2 to generate the variables necessary to fill in the PUS-C packet:

<table>
<thead>
<tr>
<th>prdID</th>
<th>N1</th>
<th>Parameter ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>str2intAgg (aggregation.name)</td>
<td>npar1</td>
<td>str2intAgg (parameterZero[].name)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NFA</th>
<th>NREP</th>
<th>N2</th>
<th>Parameter ID##</th>
</tr>
</thead>
<tbody>
<tr>
<td>nfa</td>
<td>nrep[]</td>
<td>npar2[]</td>
<td>str2intAgg (parameterNotZero[][]).name</td>
</tr>
</tbody>
</table>

### 3.2.4.3 [3,31] and [3,32] - (TC) Modify the Collection Interval of some Housekeeping/Diagnostic Parameter Report Definitions

(where 31 is for Housekeeping and 32 is for Diagnostic)

In PUS-C, the Telecommand packet, application data for the report 31 and 32, is:

<table>
<thead>
<tr>
<th>periodicFilter</th>
<th>ThresholdFilter</th>
<th>NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
</tbody>
</table>

(Where `n` is the index of a total of `N1` elements)
(Where `i` is the index of a total of `NFA` elements)
(Where `o` is the index of a total of `N2` elements)
The parameters description for this packet were omitted because they were already mentioned in the previous section 3.2.4.1.

For this request there is no direct equivalent in M&C, but it is possible to execute the same action with the updateDefinition operation.

### 3.2.4.3.1 PUS packet to M&C

In M&C, one shall use the `updateDefinition()` operation to modify the collection interval of a specific definition.

To execute the equivalent PUS-C request, one needs to perform a series of operations. One shall start by getting the corresponding object instance identifiers of the definitions:

```plaintext
definitionId = listDefinition (int2strAgg ( List<prdID[1] ... prdID[N]> ))
```

After, one shall retrieve the respective aggregation definitions from the archive:

```plaintext
aggregation = retrieve ( AggregationDefinition & definitionId )
```

The `aggregation` definition object shall now be changed. Empty fields shall remain with the previous information of the original aggregation:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td></td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td></td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td>Collection Interval[i]*</td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td><strong>parameterSets</strong></td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td><code>domain</code></td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td><code>parameters</code></td>
<td>List<a href="">MAL::Long</a></td>
<td></td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td><strong>periodicFilter</strong></td>
<td>ThresholdFilter</td>
<td></td>
</tr>
</tbody>
</table>
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

### Master Thesis

<table>
<thead>
<tr>
<th>thresholdType</th>
<th>ThresholdType</th>
<th>MAL::Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>thresholdValue</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Where \(i\) is the index of a total of \(N\) elements)

After editing the aggregation definition, one can submit it with the following operation:

**updateDefinition** (aggregation)

#### 3.2.4.3.2 M&C to PUS packet

In packet can only be generated on the very restricted case where the updateInterval field of some aggregations is edited and are submitted with the **updateDefinition** operation.

**updateDefinition** (aggregation)

It is possible to generate the following PUS-C packet:

<table>
<thead>
<tr>
<th>N</th>
<th>parameter report definition ID</th>
<th>Collection Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>count (aggregation)</td>
<td><code>str2intAgg (aggregation[].name)</code></td>
<td><code>aggregation[].updateInterval / &lt;DIAG_MIN_INTERV&gt;</code></td>
</tr>
</tbody>
</table>

#### 3.2.4.4 [3,33] and[3,34] - (TC) Report the Periodic Generation Properties of some Housekeeping/Diagnostic Parameter Report Definitions

(where 33 is for Housekeeping and 34 is for Diagnostic)

In PUS-C, the Telecommand packet, application data for the report 33 and 34, is:

```plaintext
repeated N times

<table>
<thead>
<tr>
<th>N</th>
<th>housekeeping/diagnostic parameter report definition ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned integer</td>
<td>enumerated</td>
</tr>
</tbody>
</table>
```

The parameters description for this packet were omitted because they were already mentioned in the previous section 3.2.4.1.

When this request is received, there is a generation of a Report containing the periodic properties of all the prdID definitions demanded. These reports will be seen in more detail further on ([3,35] and [3,36]).

#### 3.2.4.4.1 PUS packet to M&C

For these two telecommands, there are no direct aggregation service operations, but it is possible to recreate the same Requests/Reports by doing a request to the archive from the COM generic service. In order to do this, one starts by using M&C service operation: **listDefinition**. This will retrieve a list of MAL::Long which
correspond to the object instance identifier definitions. And then, use this same list to query the COM archive using operation \texttt{retrieve} from COM:

\begin{verbatim}
retrieve ( AggregationDefinition &
  listDefinition ( int2strAgg ( List<prdID[1] ... prdID[N] ) ) )
\end{verbatim}

<table>
<thead>
<tr>
<th>Operation</th>
<th>listDefinition</th>
<th>retrieve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input type</td>
<td>List<a href="">MAL::Identifier</a></td>
<td>ObjectType &amp; List<a href="">MAL::Long</a></td>
</tr>
<tr>
<td>Input Data</td>
<td>( \text{int2strAgg (prdID[1])} )</td>
<td>AggregationDefinition</td>
</tr>
<tr>
<td></td>
<td>( \text{int2strAgg (prdID[2])} )</td>
<td>( \text{int2strAgg (prdID[N])} )</td>
</tr>
<tr>
<td>Output type</td>
<td>List<a href="">MAL::Long</a></td>
<td>List&lt;AggregationDefinition&gt;</td>
</tr>
</tbody>
</table>

### 3.2.4.4.2 M&C to PUS packet

In M&C, the definitions are stored in the archive of the COM services, so this operation shouldn’t ever be necessary or used. Although, if one intends to get the reports [3,35] and [3,36] for some M&C aggregations, it is necessary to do the mapping.

Considering that the \texttt{aggregationDefinition} object contains all the aggregations to be requested, then the corresponding PUS-C packet would be:

\begin{verbatim}
N (aggregationDefinition) count
prdID strzintAgg (aggregationDefinition[].name)
\end{verbatim}

### 3.2.4.5 [3,35] and [3,36] - (TM) Housekeeping Parameter Report Definition

Periodic Generation Properties Report

(where 35 is for Housekeeping and 36 is for Diagnostic)

In PUS-C, the Telemetry source packet, source data for the report 35 and 36, is:

\begin{verbatim}
N
\end{verbatim}

\begin{verbatim}
housekeeping /diagnostic parameter report definition ID
periodic generation action status
collection interval
\end{verbatim}

\begin{verbatim}
unsigned integer
enumerated
enumerated
unsigned integer
\end{verbatim}

Some parameter descriptions for this packet were omitted because they were already mentioned in the previous section 3.2.4.1.
periodic generation action status defines the status of the periodic generation for the parameter report definition ID. Can be enabled or disabled.

collection interval is the collection interval for the parameter report definition ID.

In M&C there is no such report because all the AggregationDefinition objects are stored in the archive. This means that it is possible to obtain the same information without interacting with the provider. This type of report is very specific while the similar listDefinition operation is more broad because outputs an AggregationDefinition which contains all the information about the aggregation.

### 3.2.4.5.1 PUS packet to M&C

For these telemetry packets, it is possible to insert its data into an AggregationDefinition structure from M&C. These telemetry packets are only generated after sub-services 33 and 34 request them. The operation retrieve from the COM book combined with the listDefinition operation, would output a list of aggregation definitions:

\[
\text{aggregation} = \text{listDefinition} ( \text{int2strAgg} ( \text{List}<\text{prdID}[1] \ldots \text{prdID}[N]> ) )
\]

If periodic generation action status is disabled then:

\[
\text{aggregation.updateInterval} = 0
\]

The corresponding M&C AggregationDefinition structure (with the periodic generation action status enabled) would be:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td>\text{int2strAgg}(\text{prdID}[\text{n}])</td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td></td>
</tr>
<tr>
<td>category</td>
<td>AggregationCategory</td>
<td></td>
</tr>
<tr>
<td>generationEnabled</td>
<td>MAL::Boolean</td>
<td></td>
</tr>
<tr>
<td>updateInterval</td>
<td>MAL::Duration</td>
<td>\text{collection interval}[\text{n}] \ast</td>
</tr>
<tr>
<td>filterEnabled</td>
<td>MAL::Boolean</td>
<td>\text{&lt;DIAG_MIN_INTERV&gt;}</td>
</tr>
<tr>
<td>filteredTimeout</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>parameterSets</td>
<td>List<a href="">MAL::AggregationReference</a></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>List<a href="">MAL::Identifier</a></td>
<td></td>
</tr>
<tr>
<td>parameters</td>
<td>List<a href="">MAL::Long</a></td>
<td></td>
</tr>
<tr>
<td>sampleInterval</td>
<td>MAL::Duration</td>
<td></td>
</tr>
<tr>
<td>periodicFilter</td>
<td>ThresholdFilter</td>
<td></td>
</tr>
<tr>
<td>thresholdType</td>
<td>ThresholdType</td>
<td></td>
</tr>
<tr>
<td>thresholdValue</td>
<td>MAL::Attribute</td>
<td></td>
</tr>
</tbody>
</table>

(Where \text{n} is the index of a total of \text{N} elements)
3.2.4.5.2 **M&C to PUS packet**

If one intends to get the reports [3,35] and [3,36] for some M&C aggregations, it is necessary to retrieve them from the archive and then map it into a PUS-C packet.

```plaintext
aggregation = retrieve ( AggregationDefinition & listDefinition (list) )
```

If `aggregation.updateInterval == 0`  
   `periodic = Disabled (0)`
else
   `periodic = Enabled (1)`

In PUS-C, the Telemetry source packet, source data for the report 35 and 36, shall be:

<table>
<thead>
<tr>
<th>N (aggregation)</th>
<th>prdID</th>
<th>periodic generation action status</th>
<th>collection interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>str2intAgg (aggregation.name)</td>
<td>periodic</td>
<td>aggregation.updateInterval / &lt;DIAG_MIN_INTERV&gt;</td>
</tr>
</tbody>
</table>

3.2.4.6 **[3,37] - (TC) Apply some Functional Parameter Reporting Configurations**

In PUS-C, the Telecommand packet, application data for the report 37, is:

- repeated N times

<table>
<thead>
<tr>
<th>configuration execution flag</th>
<th>N</th>
<th>functional parameter reporting definition ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated</td>
<td>unsigned integer</td>
<td>enumerated</td>
</tr>
</tbody>
</table>

**Configuration execution flag** indicates whether the execution of that request is exclusive or non-exclusive. In exclusive mode, the periodic generation of all parameter reports is disabled prior to apply the Functional parameter reporting configuration.
### 3.2.4.7 [3,38] - (TC) Create a Functional Parameter Reporting Definition

In PUS-C, the Telecommand packet, application data for the report 38, is:

```
<table>
<thead>
<tr>
<th>repeated N1 times</th>
</tr>
</thead>
<tbody>
<tr>
<td>functional parameter reporting definition ID</td>
</tr>
<tr>
<td>application process ID</td>
</tr>
<tr>
<td>N2</td>
</tr>
<tr>
<td>parameter report definition ID</td>
</tr>
<tr>
<td>collection interval</td>
</tr>
</tbody>
</table>
```

functional parameter reporting definition ID: enumerated
application process ID: unsigned integer
N2: enumerated
parameter report nature: unsigned integer
parameter report definition ID: enumerated
periodic generation action status: enumerated
collection interval: unsigned integer

### 3.2.4.8 [3,39] - (TC) Delete some Functional Parameter Reporting Definitions

In PUS-C, the Telecommand packet, application data for the report 39, is:

```
<table>
<thead>
<tr>
<th>repeated N times</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
<tr>
<td>functional parameter reporting definition ID</td>
</tr>
</tbody>
</table>
```

N: unsigned integer
functional parameter reporting definition ID: enumerated

### 3.2.4.9 [3,40] - (TC) Report some Functional Parameter Reporting Definitions

In PUS-C, the Telecommand packet, application data for the report 40, is:

```
<table>
<thead>
<tr>
<th>repeated N times</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
<tr>
<td>functional parameter reporting definition ID</td>
</tr>
</tbody>
</table>
```

N: unsigned integer
functional parameter reporting definition ID: enumerated
### 3.2.4.10 [3,41] - (TM) Functional Parameter Reporting Definition Report

In PUS-C, the Telecommand packet, application data for the report 41, is:

```plaintext
repeated $N_1$ times

<table>
<thead>
<tr>
<th>functional parameter reporting definition ID</th>
<th>$N_1$</th>
<th>application process ID</th>
<th>$N_2$</th>
<th>parameter report nature</th>
<th>parameter report definition ID</th>
<th>periodic generation action status</th>
<th>collection interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated</td>
<td></td>
<td>integer</td>
<td></td>
<td></td>
<td>enumerated</td>
<td>enumerated</td>
<td>unsigned integer</td>
</tr>
</tbody>
</table>

```

### 3.2.4.11 [3,42] - (TC) Add some Parameter Report Definitions to a Functional Parameter Reporting Definition

In PUS-C, the Telecommand packet, application data for the report 42, is:

```plaintext
repeated $N_1$ times

<table>
<thead>
<tr>
<th>functional parameter reporting definition ID</th>
<th>$N_1$</th>
<th>application process ID</th>
<th>$N_2$</th>
<th>parameter report nature</th>
<th>parameter report definition ID</th>
<th>periodic generation action status</th>
<th>collection interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated</td>
<td></td>
<td>integer</td>
<td></td>
<td></td>
<td>enumerated</td>
<td>enumerated</td>
<td>unsigned integer</td>
</tr>
</tbody>
</table>

```
3.2.4.12 [3,43] - (TC) Remove some Parameter Report Definitions from a Functional Parameter Reporting Definition

In PUS-C, the Telecommand packet, application data for the report 43, is:

<table>
<thead>
<tr>
<th>functional parameter reporting definition ID</th>
<th>N1</th>
<th>application process ID</th>
<th>N2</th>
<th>parameter report definition ID</th>
<th>parameter report nature</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated</td>
<td>enumerated</td>
<td>unsigned integer</td>
<td>enumerated</td>
<td>unsigned integer</td>
<td>enumerated</td>
</tr>
</tbody>
</table>

repeated $N_1$ times

optional

3.2.4.13 [3,44] - (TC) Modify the Periodic Generation Properties of some Parameter Report Definitions of a Functional Parameter Reporting Definition

In PUS-C, the Telecommand packet, application data for the report 44, is:

<table>
<thead>
<tr>
<th>functional parameter reporting definition ID</th>
<th>N1</th>
<th>application process ID</th>
<th>N2</th>
<th>parameter report definition ID</th>
<th>parameter report nature</th>
<th>periodic generation action status</th>
<th>collection interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerated</td>
<td>enumerated</td>
<td>unsigned integer</td>
<td>enumerated</td>
<td>unsigned integer</td>
<td>enumerated</td>
<td>enumerated</td>
<td>unsigned integer</td>
</tr>
</tbody>
</table>

repeated $N_1$ times

optional
3.3 Comparison Summary

The comparison summary is sorted by PUS services (service number) and in each of these sections it includes a summary table with the corresponding MO operations and objects (the yellow color on the right means that it is using an operation from the COM services book). After, it includes the advantages and disadvantages of the usage of the CCSDS MO equivalent service. These are written in a list format sorted by different categories starting from the advantages, the neutrals facts, and continuing with the disadvantages and limitations found during the development of the systematic mapping. The advantages are represented by a green check figure. The yellow mark figure refers to a neutral fact which is not a particularly a deficiency or advantage but still should be acknowledged and in some cases even modified or changed. The red cross represents a deficiency or limitation. It is followed by a solution and the action taken in order to fix it.

3.3.1 [1] Telecommand Verification Service (COM: Activity)

Summary table with the PUS packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>MO COM Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TM) Telecommand Acceptance Report – Success</td>
<td>Event::monitorEvent() Acceptance</td>
</tr>
<tr>
<td>2</td>
<td>(TM) Telecommand Acceptance Report – Failure</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TM) Telecommand Execution Started Report – Success</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(TM) Telecommand Execution Started Report – Failure</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(TM) Telecommand Execution Progress Report – Success</td>
<td>Event::monitorEvent() Execution</td>
</tr>
<tr>
<td>6</td>
<td>(TM) Telecommand Execution Progress Report – Failure</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(TM) Telecommand Execution Completed Report – Success</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(TM) Telecommand Execution Completed Report – Failure</td>
<td></td>
</tr>
</tbody>
</table>

The TM[1,9] doesn’t exist because it was not standardized in the PUS-C. The decision of not using TM[1,9] in PUS-C was chosen because someone might like to use that message type on the future (it would be: “successful routing verification report”).

Summary table with the PUS-C packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO COM Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>(TM) Failed Routing Verification Report</td>
<td>Event::monitorEvent() Reception</td>
</tr>
</tbody>
</table>

a. All the PUS service 1 reports, have 2 fields which are not covered in MO, the Telecommand Packet ID and the Packet Sequence Control. This happens because the concept of packets no longer exists in MO, so a direct equivalent field is not possible.✓

b. In PUS, there’s an optional Parameter field which provides complementary information relating to the particular value of the Code field. An equivalent field also does not exist in the MO Activity structures. ●
Suggested solution: This field is optional, so it can be omitted. A second solution could be achieved by including one more field in all the three MO COM Activity structures.

Action performed: RIDs to fix this deficiency submitted by Brigitte Behal.

c. In PUS, when a failure is reported, there is always a Code field which indicates the reason for that failure. In MO, the equivalent generated events do not include any type of information regarding the error. ✗

Suggested solution: include an errorInformation field in the structures: ActivityTransfer, ActivityAcceptance and ActivityExecution. It would be nullable if the success field is TRUE. If the success field was FALSE it would contain an enumeration of Errors (check PUS page 53).

Action performed: RIDs to fix this deficiency submitted by Brigitte Behal.

d. In PUS, there is no stageCount equivalent field (ActivityExecution.stageCount: The total number of execution stages that will be reported). An unknown number of stages are not allowed to be represented in the ActivityExecution structure. ✗

Suggested solution: Make the stageCount field nullable and define it as: “If NULL, the total number of execution stages is unknown.”

When the stageCount is NULL, to represent the final execution stage, one needs to make the executionStage nullable in order to represent this exceptional case.

Action performed: RID_522-1-R-3_CC_01 (section: 3.4.1) submitted by César Coelho.

3.3.2 [3] Housekeeping and Diagnostic Data Reporting Service (M&C: Aggregation)

Summary table with the PUS packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>MO M&amp;C Aggregation operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Define New Housekeeping Parameter Report</td>
<td>addDefinition()</td>
</tr>
<tr>
<td>2</td>
<td>(TC) Define New Diagnostic Parameter Report</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Clearing Housekeeping Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(TC) Clearing Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(TC) Enable Housekeeping Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(TC) Disable Housekeeping Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(TC) Enable Diagnostic Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(TC) Disable Diagnostic Parameter Report Generation</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(TC) Report Housekeeping Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(TM) Housekeeping Parameter Report Definitions Report</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(TC) Report Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(TM) Diagnostic Parameter Report Definitions Report</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(TC) Report Housekeeping Parameter Sampling-Time Offsets</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(TC) Report Diagnostic Parameter Sampling-Time Offsets</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(TM) Housekeeping Parameter Sampling-Time Offsets Report</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(TM) Diagnostic Parameter Sampling-Time Offsets Report</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(TC) Select Periodic Housekeeping Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>(TC) Select Periodic Diagnostic Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>(TC) Select Filtered Housekeeping Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>(TC) Select Filtered Diagnostic Parameter Report Generation Mode</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>(TC) Report Unfiltered Housekeeping Parameters</td>
<td></td>
</tr>
</tbody>
</table>
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO M&amp;C Aggregation operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>(TC) Report Unfiltered Diagnostic Parameters</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>(TM) Unfiltered Housekeeping Parameters Report</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>(TM) Unfiltered Diagnostic Parameters Report</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>(TM) Housekeeping Parameters Report</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>(TM) Diagnostic Parameters Report</td>
<td>monitorValue()</td>
</tr>
</tbody>
</table>

Summary table with the PUS-C packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO M&amp;C Aggregation operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>(TC) Generate one shot Report for some Housekeeping Parameter Report Definitions</td>
<td>getValue()</td>
</tr>
<tr>
<td>28</td>
<td>(TC) Generate one shot Report for some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>(TC) Append Parameters to a Housekeeping Parameter Report Definition</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>30</td>
<td>(TC) Append Parameters to a Diagnostic Parameter Report Definition</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>(TC) Modify the Collection Interval of some Housekeeping Parameter Report Definitions</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>32</td>
<td>(TC) Modify the Collection Interval of some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>(TC) Report the Periodic Generation Properties of some Housekeeping Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>(TC) Report the Periodic Generation Properties of some Diagnostic Parameter Report Definitions</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>(TM) Housekeeping Parameter Report Definition Periodic Generation Properties Report</td>
<td>listDefinition()</td>
</tr>
<tr>
<td>36</td>
<td>(TM) Diagnostic Parameter Report Definition Periodic Generation Properties Report</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>(TC) Apply some Functional Parameter Reporting Configurations</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>(TC) Create a Functional Parameter Reporting Definition</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>(TC) Delete some Functional Parameter Reporting Definitions</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>(TC) Report some Functional Parameter Reporting Definitions</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>(TM) Functional Parameter Reporting Definition Report</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>(TC) Add some Parameter Report Definitions to a Functional Parameter Reporting Definition</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>(TC) Remove some Parameter Report Definitions from a Functional Parameter Reporting Definition</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>(TC) Modify the Periodic Generation Properties of some Parameter Report Definitions from a Functional Parameter Reporting Definition</td>
<td></td>
</tr>
</tbody>
</table>

a. Group concept doesn’t exist in PUS. Although, it is possible to map the MO instructions which use the group concept into PUS packets by cycling through all the objects within that group.

b. In PUS, the data collection interval is restricted to a multiple of the <DIAG_MIN_INTERV> unit. M&C allows to define the equivalent updateInterval (type MAL::Duration) with a higher precision because a MAL::Duration value, “represents a length of time in seconds. It may contain a fractional component”.

c. “Select Periodic Report Generation mode” in PUS ([3,17] and [3,18]), destroy the previous
configurations set for the filtered mode, which means that every time one wants to activate the filtered mode, one has to send all the configuration parameters for the filter once again (check [3,19] and [3,20]). In M&C, one can simply enable/disable the periodic mode without destroying the filter configurations (using the enableFilter operation).

d. In PUS, the filter can be defined per parameter per definition. In M&C it is possible to select a different filter for every parameterSets within the aggregation definition as long as it contains only one parameter per AggregationReference (4.7.6 periodicFilter). If the AggregationReference contains more than one parameter, then it shall not be possible to select a filter.

e. In PUS-C, there is a new subservice named “parameter reporting configuration subservice” which provides the capability to control the generation of the parameter reports generated by the housekeeping and the diagnostic reporting subservices. MO does not have an equivalent subservice.

Suggested solution: Define 2 new objects to represent the different configurations and add a set of new operations to manage these configurations.

Action performed: RID_522-1-R-3_CC_02 (section:3.4.2) submitted by César Coelho.

3.3.3 [4] Parameter Statistics Reporting Service (M&C: Statistic)

Summary table with the PUS packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>MO M&amp;C Statistic operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Report Parameter Statistics</td>
<td>getStatistics()</td>
</tr>
<tr>
<td>2</td>
<td>(TM) Parameter Statistics Report</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Reset Parameter Statistics Reporting</td>
<td>resetEvaluation()</td>
</tr>
<tr>
<td>4</td>
<td>(TC) Enable Periodic Parameter Statistics Reporting</td>
<td>updateParameterEvaluation()</td>
</tr>
<tr>
<td>5</td>
<td>(TC) Disable Periodic Parameter Statistics Reporting</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(TC) Add Parameters to Parameter Statistics List</td>
<td>addParameterEvaluation()</td>
</tr>
<tr>
<td>7</td>
<td>(TC) Delete Parameters from Parameter Statistics List</td>
<td>removeParameterEvaluation()</td>
</tr>
<tr>
<td>8</td>
<td>(TC) Report Parameter Statistics List</td>
<td>getStatistics()</td>
</tr>
<tr>
<td>9</td>
<td>(TM) Parameter Statistics List Report</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(TC) Clear Parameter Statistics List</td>
<td>removeParameterEvaluation()</td>
</tr>
</tbody>
</table>

a. In M&C, the operation getStatistics allows the user to select specific statistic functions and specific parameter objects. In PUS, the equivalent request [4,1], doesn’t provide that selection, and so the request will always generate a full report with all the Parameter Statistics and all their respective functions.

b. In PUS, for the telecommand [4,2] it is possible to report one particular type of evaluation once per parameter. In M&C it is possible to report multiple evaluations per parameter (StatisticParameterReport.values.value is of type List<StatisticValue>).
c. The PUS telecommand [4,2] reports the current parameter statistic values for four different functions, respectively, Maximum, Minimum, Mean and an optional Standard Deviation. The equivalent M&C StatisticParameterReport structure report is more flexible and allows the reporting of any number of functions previously defined (not just restricted to a number of 4 like PUS).

d. In PUS, the reset parameters telecommand [4,3] does not allow the reset of each parameter individually. In M&C, it is possible to reset all the parameters at the same time, or each one individually, or a group of them, with the operation \texttt{resetEvaluation}.

e. In PUS, there is a limitation of four Statistic evaluations per parameter and the Statistic List is organized by parameter (parameter dependency). In M&C, it can exist different StatisticParameter objects for the same parameter (no parameter dependency).

f. In PUS, the Reporting Interval can only be selected to all the Parameter Statistics simultaneously. In M&C, it is possible to have a different Reporting Interval for each Statistic Link.

g. In PUS, the Reporting Interval field sets the interval of time for the periodic reporting and resetting of the parameter statistics. In M&C, these two actions are separated: the interval of time for the periodic reporting is defined by the reportingInterval field and the resetting of the parameter statistics is defined by the collectionInterval field.

### 3.3.4 [5] Event Reporting Service (M&C: Alert)

Summary table with the PUS packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>MO M&amp;C Alert operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TM) Normal/Progress Report</td>
<td>Event::monitorEvent()</td>
</tr>
<tr>
<td>2</td>
<td>(TM) Error/Anomaly Report - Low Severity</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TM) Error/Anomaly Report - Medium Severity</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(TM) Error/Anomaly Report - High Severity</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(TC) Enable Event Report Generation</td>
<td>enableGeneration()</td>
</tr>
<tr>
<td>6</td>
<td>(TC) Disable Event Report Generation</td>
<td></td>
</tr>
</tbody>
</table>

Summary table with the PUS-C packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO M&amp;C Alert operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>(TC) Report the list of Disabled Event Reports</td>
<td>listDefinition()</td>
</tr>
<tr>
<td>8</td>
<td>(TM) Disabled Event Report List Report</td>
<td></td>
</tr>
</tbody>
</table>
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

a. In PUS, there are different 4 types of report where which one of them represents a different degree of severity ( [5,1] Normal; [5,2] Low Severity; [5,3] Medium Severity; [5,4] High Severity). In M&C, there are 5 different enumerations for the severity field (INFORMATIONAL, WARNING, ALARM, SEVERE, CRITICAL).

3.3.5 [12] On-board Monitoring Service (M&C: Check)

Summary table with the PUS packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS number</th>
<th>PUS Request/Report</th>
<th>MO M&amp;C Check operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Enable Monitoring of Parameters</td>
<td>enableCheck()</td>
</tr>
<tr>
<td>2</td>
<td>(TC) Disable Monitoring of Parameters</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Change Maximum Reporting Delay</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>4</td>
<td>(TC) Clear Monitoring List</td>
<td>removeDefinition()</td>
</tr>
<tr>
<td>5</td>
<td>(TC) Add Parameters to Monitoring List</td>
<td>addParameterCheck()</td>
</tr>
<tr>
<td>6</td>
<td>(TC) Delete Parameters from Monitoring List</td>
<td>removeDefinition()</td>
</tr>
<tr>
<td>7</td>
<td>(TC) Modify Parameter Checking Information</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>8</td>
<td>(TC) Report Current Monitoring List</td>
<td>listDefinition()</td>
</tr>
<tr>
<td>9</td>
<td>(TM) Current Monitoring List Report</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(TM) Check Transition Report</td>
<td>Event::monitorEvent()</td>
</tr>
</tbody>
</table>

Summary table with the PUS-C packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO M&amp;C Check/Parameter operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Enable some Parameter Monitoring Definitions</td>
<td>enableCheck()</td>
</tr>
<tr>
<td>2</td>
<td>(TC) Disable some Parameter Monitoring Definitions</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Change the Maximum Transition Reporting Delay</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>4</td>
<td>(TC) Delete all Parameter Monitoring Definitions</td>
<td>removeDefinition()</td>
</tr>
<tr>
<td>5</td>
<td>(TC) Add some Parameter Monitoring Definitions</td>
<td>addDefinition()</td>
</tr>
<tr>
<td>6</td>
<td>(TC) Delete some Parameter Monitoring Definitions</td>
<td>addParameterCheck()</td>
</tr>
<tr>
<td>7</td>
<td>(TC) Modify some Parameter Monitoring Definitions</td>
<td>removeDefinition()</td>
</tr>
<tr>
<td>8</td>
<td>(TC) Report some Parameter Monitoring Definitions</td>
<td>updateDefinition()</td>
</tr>
</tbody>
</table>
In PUS, a check definition shall be “enabled” and used for checking a parameter when all the following conditions are set:

1. The monitoring of parameters is enabled at service level;
2. The monitoring of parameters is enabled;
3. The parameter is valid;
4. Check definition is selected for checking the sample (its Check Selection Parameter == TRUE).

a. In PUS for one specific check definition, there is only 1 parameter connected. In M&C, it is possible to have multiple parameters connected to the same check definition. To have the same behavior in PUS, it is necessary to redefine the check definition #parameters times. ✓

b. In M&C, it is possible to have a different maxReportingInterval per each check definition, and its value can be modified using the `updateDefinition` operation. PUS only allows the control over one Max Reporting Delay for all the definitions (complete service). It is also important to mention that Max Reporting Delay can only be expressed as a multiple of `<DIAG_MIN_INTERV>` while the equivalent maxReportingInterval has a MAL::Duration type which can “represent a length of time in seconds. It may contain a fractional component”. ✓

c. In M&C, the `getCurrentTransitionList` operation request has more functionality than its equivalent PUS telecommand [12,10] and telemetry [12,11] packets. The main difference is that it allows the input of a `CheckResultFilter` object which defines a filter for the returned check
d. The constantCheck object allows the checking against a list of values. In PUS, the comparison is only performed on a single value per definition. To perform the same in PUS, it would be necessary to create multiple definitions. ✓

e. In PUS, the Parameter Monitor Interval is restricted to a multiple of the `<DIAG_MIN_INTERV>` unit. M&C allows to define the equivalent checkInterval (type MAL::Duration) with a higher precision because a MAL::Duration value, “represents a length of time in seconds. It may contain a fractional component”. ✓

f. In PUS and PUS-C, there is no equivalent for the violationCount field from the CheckDefinition object (“Number of consecutive valid samples violating the check to be in violation. Must be greater than zero”). ✓

g. For the second condition, the equivalent representation used in this document M&C was the checkEnabled field of the CheckLink object. This field can be changed by using the enableCheck operation. In PUS, to enable/disable a certain parameter, TC[12,1] and TC[12,2] shall be used. It is important to understand that this is not an exact equivalent because in PUS, the enabling/disabling occurs at the parameter level while in M&C it occurs at the CheckLink level. The mapping is allowed but with some restrictions.

h. In PUS, when more than one check definition is defined for a given parameter, each one has a different “check position” which is determined by its position in the request (however, the check position may be changed by a subsequent “modify parameter checking information” request). In M&C, this indexation of check lists per parameter does not exist.

Suggested solution: In M&C it is possible to take advantage of the description field of the CheckDefinition object to store the “equivalent” PUS index per parameter. The solution is far from perfect but it allows the mapping. As an example, for the first check link element of the PUS for a given parameter which contains a total of 4 check links, the description field could be:

```
CheckDefinition.description = “1 of 4”
```

i. PUS and M&C have different ways of representing the Checking Status. An equivalence table follows (table1). ✓

<table>
<thead>
<tr>
<th>PUS Checking Status value</th>
<th>PUS Checking Status</th>
<th>M&amp;C equivalent check state</th>
<th>M&amp;C equivalent evaluation result</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>“Expected Value”, “Within Limits”, “Within Thresholds”</td>
<td>OK 4</td>
<td>NULL</td>
</tr>
<tr>
<td>1</td>
<td>“Unchecked”</td>
<td>DISABLED 1</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>“Invalid”</td>
<td>INVALID 3</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>“Unselected”</td>
<td>UNCHECKED 2</td>
<td>NULL</td>
</tr>
</tbody>
</table>
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

<table>
<thead>
<tr>
<th></th>
<th>“Unexpected Value”, “Below Low Limit”, “Below Low Threshold”</th>
<th>NOT_OK 5</th>
<th>FALSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>“Above High Limit”, “Above High Threshold”</td>
<td>NOT_OK 5</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

j. PUS-C and M&C have different ways of representing the Checking Status. An equivalence table follows (table2).

<table>
<thead>
<tr>
<th>Table2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PUS-C Checking Status raw value</strong></td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

k. For the first condition above, there is no equivalent in M&C. The operation `enableCheck` with the wildcard selected will disable all the check links on the provider. This is not the same as disabling the monitoring of Parameters at a service level because, for example, if one intends to activate back the PUS Monitoring List at the service level, all the check links will still hold the previous state before the commutation of state of the monitoring list at service level, while in M&C the state of all the check links will be switched to 0 when the `enableCheck` operation is used. 🗿

Suggested solution: change the `enableCheck` operation to enable/disable at a service level when used with the wildcard 0. Or, create a new object to represent the status of the Monitoring List:

<table>
<thead>
<tr>
<th>CheckMonitoringList</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>description</td>
</tr>
<tr>
<td>listEnabled</td>
</tr>
<tr>
<td>functionalMonitoring</td>
</tr>
</tbody>
</table>

Action performed: RID_522-1-R-3_CC_12 (section: 3.4.10) submitted by César Coelho.

l. In PUS-C the concept of “Functional Monitoring Lists” is introduced. The Functional Monitoring
subservice provides the capability for controlling the functional monitoring function and controlling the functional monitoring definitions. In M&C, there is no equivalent of a Functional Monitoring list.

**Suggested solution:** It is necessary to redefine a set of objects and operations in order to have the same behavior. The concept of multiple Monitoring Lists could be implemented in M&C by defining the CheckMonitoringList proposed in k. in combination with the following FunctionalMonitoring object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>monitoringList</td>
<td>List<a href="">MAL::Long</a></td>
<td>No</td>
<td>The list shall contain the object instance identifiers of the check definitions associated with the functional monitoring list.</td>
</tr>
<tr>
<td>protected minimumFailures</td>
<td>MAL::Boolean</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>condition</td>
<td>ParameterExpression</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>parameterId</td>
<td>COM::ObjectKey</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>operator</td>
<td>COM::ExpressionOperator</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>useConverted</td>
<td>MAL::Boolean</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>Mal::Attribute</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

The operations to be added: listMonitoringList, addMonitoringList, removeMonitoringList, enableMonitoringList, protectMonitoringList, getMonitoringListReportStatus.

**Action performed:** RID_522-1-R-3_CC_13 (section: 3.4.11) submitted by César Coelho.

m. Requirement 3.5.2.33: “The maxReportingInterval value may be set by using the updateDefinition operation”, cannot be optional because the equivalent action in PUS is not optional: Telecommand [12,3] “Change Maximum Reporting Delay”. **✗**

**Action performed:** RID_522-1-R-3_CC_04 (section: 3.4.3) submitted by César Coelho.

### 3.3.6 [20] On-board Parameter Management Service (M&C: Parameter)

Summary table with the PUS-C packets matching the MO operations:

<table>
<thead>
<tr>
<th>PUS-C number</th>
<th>PUS-C Request/Report</th>
<th>MO M&amp;C Parameter operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(TC) Report some Parameter Values</td>
<td>getValue()</td>
</tr>
<tr>
<td>2</td>
<td>(TM) Parameter Values Report</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(TC) Set some Parameter Values</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(TC) Change some Raw Memory Parameter Definitions</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>5</td>
<td>(TC) Change some Object Memory Parameter Definitions</td>
<td>updateDefinition()</td>
</tr>
<tr>
<td>6</td>
<td>(TC) Report some Parameter Definitions</td>
<td>listDefinition()</td>
</tr>
</tbody>
</table>
a. In M&C it is possible to enable/disable the generation of updates for a particular parameter with the `enableGeneration` operation. In PUS-C, the option to enable/disable the generation of parameter updates does not exist. ✓

b. The PUS-C packet request [20,3] (“set some parameter values”) cannot be mapped into a M&C operation because there is no MO equivalent operation which allows the user to set a parameter value. ✓


Action performed: RID_522-1-R-3_CC_07 (section: 3.4.6) submitted by César Coelho.

c. The PUS-C packet requests [20,4], [20,5] and [20,7] cannot be mapped into a M&C operation because the ParameterDefinition object does not have an equivalent field for the memory ID field, for the address field and for the base/offset. ✗

Suggested solution: Add three new fields in the ParameterDefinition structure: `memoryId`; `base`; `address`. The `memoryId` field would hold the PUS memory ID. The `base` field could be nullable and that would determine the addressing scheme. The `address` field would hold the absolute address or an offset depending on the base being define to some value or set to NULL.

Action performed: RID_522-1-R-3_CC_08 (section: 3.4.7) submitted by César Coelho.

3.3.7 Others

n. “The list may contain the wildcard value of 0” (pages 37, 49, 58, 75, 77, 92, 105 of the CCSDS 522.1-R-3 M&C book). These requirements cannot be optional. The words “may contain” need to be replaced by “shall support”. ✗

Action performed: Email sent directly to Sam Cooper.

o. Suggestion of a relationship diagram on the Check Service. ✓

Action performed: RID_522-1-R-3_CC_09 (section: 3.4.8) submitted by César Coelho.

p. The Page numbering style is not very good because the “pdf” page does not match the physical page. ✓

Action performed: RID_522-1-R-3_CC_10 (section: 3.4.9) submitted by César Coelho.

q. The severity of an AlertEvent should be directly stamped on the AlertDefinition object. ✓

Action performed: RID_522-1-R-3_CC_05 (section: 3.4.4) submitted by César Coelho.

r. Some editorial errors were found during the generation of this document. ✗

Action performed: RID_522-1-R-3_CC_06 (section: 3.4.5) submitted by César Coelho. Previously, the errors were also reported to Sam Cooper via email.
3.4 Review Item Disposition (RID)

3.4.1 RID 01

REVIEW ITEM DISPOSITION (RID):
RED BOOK RID INITIATION FORM

AGENCY RID NUMBER: CC_01
SUBMITTING ORGANIZATION (Agency, Center): ESA
------------------------------------------------------------------
REVIEWER’S NAME: César Coelho
CODE:
E-MAIL ADDRESS: Cesar.Coelho@esa.int
TELEPHONE: -
------------------------------------------------------------------
DOCUMENT NUMBER: CCSDS 521.1-B-1 Blue Book, Issue 1
DOCUMENT NAME: MO Common Object Model
DATE ISSUED: February 2014
PAGE NUMBER: 4-11 (page 60)
PARAGRAPH NUMBER: Table (stageCount, executionStage)
RID SHORT TITLE: Add the possibility to represent an unknown number of execution stages
------------------------------------------------------------------
DESCRIPTION OF REQUESTED CHANGE: (Use From: "..." To "...") format
PUS compliance

In the ActivityExecution structure table, the stageCount field would have to be nullable. The comment field shall state: “The total number of execution stages that will be reported. If NULL then the total number of execution stages is Unknown.”

To represent the final execution stage (stageCount == executionStage) when the number of stages is Unknown, it will be necessary to make the field executionStage Nullable, as well. The comment field shall state: “The execution stage of the operation. If NULL then it represents the final execution stage of an unknown total number of execution stages.”

CATEGORY OF REQUESTED CHANGE:

Technical Fact ___ X ___ Recommended ___ Editorial ___

NOTES:

TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)
RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.
EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

SUPPORTING ANALYSIS:
PUS does not report the total number of execution stages.
This fact can be verified by checking any of the packets [1,3], [1,4], [1,5], [1,6], [1,7], [1,8]. (page 53, 54)
If the total number of execution stages is not reported, it means that the number of execution stages is unknown to any user without prior knowledge of the mission details.
MO M&C does not allow the representation of an unknown number of execution stages.
------------------------------------------------------------------
DISPOSITION:
3.4.2 RID 02

**REVIEW ITEM DISPOSITION (RID):**
RED BOOK RID INITIATION FORM

**AGENCY RID NUMBER:** CC_02
**SUBMITTING ORGANIZATION (Agency, Center):** ESA

**REVIEWER’S NAME:** César Coelho
**CODE:**
**E-MAIL ADDRESS:** Cesar.Coelho@esa.int
**TELEPHONE:**

**DOCUMENT NUMBER:** CCSDS 522.1-R-3  Red Book, Issue 3
**DOCUMENT NAME:** MO Monitor and Control Services
**DATE ISSUED:** March 2014
**PAGE NUMBER:** 3-69 (page 94)  **PARAGRAPH NUMBER:** Requirements

**RID SHORT TITLE:** Add Configurations in the Aggregation Service

**DESCRIPTION OF REQUESTED CHANGE:**

PUS-C compliance

It shall be possible to apply, add, remove and modify a specific configuration. 5 new operations need to be defined, e.g., listConfiguration, addConfiguration, updateConfiguration, removeConfiguration and applyConfiguration.

3.7.2.1: The aggregation service shall provide:
e) the capability for applying, managing, reporting and reporting the list of aggregation configurations.

To be able to use these operations and to fulfil the requirement, it would be necessary to make an AggregationConfiguration object and an AggregationConfigurationInfo object which would contain the configuration data. Suggestion:

AggregationConfiguration with 3 fields: name (type: MAL::Identifier, nullable:no); description (type: MAL::String, nullable:yes); definitionSets (type: List<AggregationConfigurationInfo>, nullable:no)
AggregationConfigurationInfo with 3 fields: aggregationId (type:MAL::Long, nullable:no); category (type:AggregationCategory, nullable:no); parameterSets (type:List<MAL::AggregationReference>, nullable:no)

**CATEGORY OF REQUESTED CHANGE:**

Technical Fact __X__  Recommended ___  Editorial ___

**NOTES:**

**TECHNICAL FACT:** Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

**RECOMMENDED:** Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

**EDITORIAL:** Typographical or other factual error needing correction.
(This type of change will be made without feedback to submitter.)

**SUPPORTING ANALYSIS:**

In PUS-C (PUS_C Draft 8[20140422]) there’s a new Parameter reporting configuration subservice (page 115: PUS_C Draft 8[20140422]) in the Housekeeping and Diagnostic Data Reporting Service. This subservice provides the capability for applying the functional parameter reporting configurations; the capability for managing/reporting/modifying the functional parameter reporting definitions.
### 3.4.3 RID 04

**REVIEW ITEM DISPOSITION (RID):**
RED BOOK RID INITIATION FORM

**AGENCY RID NUMBER:** CC_04

**SUBMITTING ORGANIZATION (Agency, Center):** ESA

**REVIEWER'S NAME:** César Coelho

**CODE:** E-

**E-MAIL ADDRESS:** Cesar.Coelho@esa.int

**TELEPHONE:** -

---

**DOCUMENT NUMBER:** CCSDS 522.1-R-3 Red Book, Issue 3

**DOCUMENT NAME:** MO Monitor and Control Services

**DATE ISSUED:** March 2014

**PAGE NUMBER:** 3-40 (page 65) **PARAGRAPH NUMBER:** 2

**RID SHORT TITLE:** Change in requirement 3.5.2.33

---

**DESCRIPTION OF REQUESTED CHANGE:**
(Use From: "..." To "..." format)

PUS compliance

The requirement 3.5.2.33 says:
"The maxReportingInterval value may be set by using the updateDefinition operation."

To make it compliant with PUS, this specification cannot be optional:
"The maxReportingInterval value shall be set by using the updateDefinition operation."

**CATEGORY OF REQUESTED CHANGE:** Technical Fact __X__ Recommended ___ Editorial ___

**NOTES:**
TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

**SUPPORTING ANALYSIS:**
In PUS, page 124:
"Change Maximum Reporting Delay (12.3)"

“When the service provider receives this request, the maximum reporting delay shall be recorded and used to determine when to downlink the transition reporting list.”

In MO M&C page 15 (1-4):
“c) the word ‘may’ implies an optional specification;”
“a) the words ‘shall’ and ‘must’ imply a binding and verifiable specification;”

**DISPOSITION:**
3.4.4 **RID 05**

**REVIEW ITEM DISPOSITION (RID):**

RED BOOK RID INITIATION FORM

**AGENCY RID NUMBER:** CC_05

**SUBMITTING ORGANIZATION (Agency, Center):** ESA

**REVIEWER'S NAME:** César Coelho

**CODE:**

**E-MAIL ADDRESS:** Cesar.Coelho@esa.int

**TELEPHONE:**

**DOCUMENT NUMBER:** CCSDS 522.1-R-3 Red Book, Issue 3

**DOCUMENT NAME:** MO Monitor and Control Services

**DATE ISSUED:** March 2014

**PAGE NUMBER:** 4-12 (page 121)

**PARAGRAPH NUMBER:** Table

**RID SHORT TITLE:** Stamp severity field in the AlertDefinition object

**DESCRIPTION OF REQUESTED CHANGE:** (Use From: "..." To "..." format)

Add a severity field in the AlertEvent object.

Comment: “Severity of the AlertEvent. Holds the severity of the alert at the time of its generation.”

**CATEGORY OF REQUESTED CHANGE:**

Technical Fact **X**  Recommended ___  Editorial ___

**NOTES:**

TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

**SUPPORTING ANALYSIS:**

M&C page 121:

“The AlertEvent structure holds an instance of an alert defined using an AlertDefinition.”

Currently, it is only possible to determine the severity of the AlertEvent by checking its related field which contains the AlertDefinition that generated the Alert. (requirement 3.4.3.3 page 51)

Errors might occur. Let’s consider the following timeline events situation:

Day 1: There’s an AlertEvent1 generated by the AlertDefinition1. The user1 checks its severity by checking the related field and it is set as INFORMATIONAL.

Day 2: The severity field of AlertDefinition1 is modified to ALARM (with operation updateDefinition) by user2.

Day 3: The user3 checks the severity of AlertEvent1 and he sees it as ALARM.

The severity of the AlertEvent1 was different for user1 and user3. The severity of a certain AlertEvent shouldn’t change. Of course this is a very exceptional case although this type of time evolution in the severity of a certain AlertEvent shouldn’t happen! It is necessary that each AlertEvent holds its own severity at the time of its generation.
3.4.5 **RID 06**

REVIEW ITEM DISPOSITION (RID):
RED BOOK RID INITIATION FORM

AGENCY RID NUMBER: CC_06
SUBMITTING ORGANIZATION (Agency, Center): ESA

REVIEWER'S NAME: César Coelho
CODE: E-MAIL ADDRESS: Cesar.Coelho@esa.int
TELEPHONE:-------------------------------------------------------------

DOCUMENT NUMBER: CCSDS 522.1-R-3 Red Book, Issue 3
DOCUMENT NAME: MO Monitor and Control Services
DATE ISSUED: March 2014
PAGE NUMBER: pages 53, 70, 98, 86, 122
PARAGRAPH NUMBER: -

RID SHORT TITLE: Some typos and mistakes in the book

DESCRIPTION OF REQUESTED CHANGE: (Use From: "..." To "...") format

Typos:
- (page 53 table) From: "COM::InstanceBooleanPair" To: "List<COM::InstanceBooleanPair>" (check 3.4.6.2.3)
- (page 53 3.4.6.2.8) From: "generation enabled" To: "generationEnabled"
- (page 70 3.5.9.2.7) From: "If a request is does not..." To: "If a request does not..."
- (page 98, 3.7.7.1) From: "eneableGeneration" To: "enableGeneration"

Mistakes:
- (page 86) The information content on 3.6.6.2.7 and on 3.6.6.2.8 is the same:
  3.6.6.2.7: “The request shall include a final Boolean indicating whether to return the current evaluation result for each of the matched links before resetting; if the value is TRUE then the current evaluation is returned; otherwise a NULL is returned.”
  3.6.6.2.8: “If the supplied Boolean is set to TRUE, the Response shall contain the latest statistic evaluations before resetting them; otherwise it returns NULL.”
- (page 122 CheckState enumeration) In the definition of NOT_OK From: "All checks not OK" To: "One or more checks are not OK"

CATEGORY OF REQUESTED CHANGE: Technical Fact ___ Recommended ___ Editorial ___ X ___

NOTES:
TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)
RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.
EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

SUPPORTING ANALYSIS:
These typos and mistakes were directly submitted to Sam via email on the 24th of April 2014. So, perhaps they were already fixed.

DISPOSITION:


**RID 07**

**REVIEW ITEM DISPOSITION (RID):**

**RED BOOK RID INITIATION FORM**

**AGENCY RID NUMBER:** CC_07

**SUBMITTING ORGANIZATION (Agency, Center):** ESA

---

**REVIEWER'S NAME:** César Coelho

**CODE:**

**E-MAIL ADDRESS:** Cesar.Coelho@esa.int

**TELEPHONE:** -

---

**DOCUMENT NUMBER:** CCSDS 522.1-R-3

**DOCUMENT NAME:** MO Monitor and Control Services

**DATE ISSUED:** March 2014

**PAGE NUMBER:** 3-12 (page 37)

**PARAGRAPH NUMBER:** RID SHORT TITLE: **Add setParameterValue operation in Parameter Service**

---

**DESCRIPTION OF REQUESTED CHANGE:** (Use From: "..." To "..." format)

PUS-C compliance

It is necessary to define a new operation to set a parameter value.

Suggestion:

`setParameterValue()`

The setParameterValue operation allows a consumer to set a parameter value for one or more parameters.

InteractionPattern: SUBMIT

**IN – SUBMIT**: `List<MAL::Long> & List<ParameterValue>`

1. The submitted lists shall hold the parameter and its respective value to be set.
2. The first list shall contain the object instance identifiers of the parameter definitions to be set.
3. The second list shall contain the parameter values to be set.
4. If a request parameter is unknown then an UNKNOWN error shall be returned.

---

**CATEGORY OF REQUESTED CHANGE:**

| Technical Fact | __X__ | Recommended | ___ | Editorial | ___ |

**NOTES:**

**TECHNICAL FACT:** Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

**RECOMMENDED:** Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

**EDITORIAL:** Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

---

**SUPPORTING ANALYSIS:**

PUS-C page 550 (PUS_C Draft 8[20140422])

Currently, there is no equivalent in MO M&C for the TC[20,3] in PUS-C (set some parameter values).

---

**DISPOSITION:**
3.4.7 **RID 08**

**REVIEW ITEM DISPOSITION (RID):**  
RED BOOK RID INITIATION FORM

**AGENCY RID NUMBER:** CC_08  
**SUBMITTING ORGANIZATION (Agency, Center):** ESA

---

**REVIEWER'S NAME:** César Coelho  
**CODE:**  
**E-MAIL ADDRESS:** Cesar.Coelho@esa.int  
**TELEPHONE:**

---

**DOCUMENT NUMBER:** CCSDS 522.1-R-3  
**DOCUMENT NAME:** MO Monitor and Control Services  
**DATE ISSUED:** March 2014  
**PAGE NUMBER:** 4-8 (page 117)  
**PARAGRAPH NUMBER:** Table

**RID SHORT TITLE:** Add the possibility to select between different logical memory spaces (Parameter service)

---

**DESCRIPTION OF REQUESTED CHANGE:** (Use From: "..." To "..." format)

PUS-C compliance

Add 3 new fields in the ParameterDefinition structure: memoryId; base; address.

- **memoryId** (Type: MAL::Long Nullable: no/yes Comment: “If the satellite hosts more than one on-board memory, then this field shall be used to identify different logical memory spaces, potentially managed by different on-board processors.”)
- **base** (Type: MAL::ULong Nullable: yes Comment: “Holds the base value of a ‘base plus offset’ addressing scheme. If NULL then the addressing scheme shall be of ‘absolute address type’.”)
- **address** (Type: MAL::ULong Nullable: yes Comment: “It shall hold an offset. If the base field is NULL then it holds the absolute address.”)

---

**CATEGORY OF REQUESTED CHANGE:**  
Technical Fact ____  
Recommended ____  
Editorial ____

**NOTES:**  
**TECHNICAL FACT:** Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

**RECOMMENDED:** Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

**EDITORIAL:** Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

---

**SUPPORTING ANALYSIS:**  
PUS-C page 551 (PUS-C Draft 8[20140422])

Currently, the TC[20,4], TC[20,5], TM[20,7] in PUS-C cannot be mapped into a MO M&C

These have three fields which do not have an equivalent in M&C: memory ID, base, offset.

---

**DISPOSITION:**
3.4.8 RID 09

REVIEW ITEM DISPOSITION (RID):
RED BOOK RID INITIATION FORM

AGENCY RID NUMBER: CC_09
SUBMITTING ORGANIZATION (Agency, Center): ESA

REVIEWER'S NAME: César Coelho
CODE: -
E-MAIL ADDRESS: Cesar.Coelho@esa.int
TELEPHONE: -

DOCUMENT NUMBER: CCSDS 522.1-R-3 Red Book, Issue 3
DOCUMENT NAME: MO Monitor and Control Services
DATE ISSUED: March 2014
PAGE NUMBER: 3-36 (page 61) PARAGRAPH NUMBER: 2
RID SHORT TITLE: Include objects relationship diagram for the Check Service

DESCRIPTION OF REQUESTED CHANGE: (Use From: "..." To "..." format)
There should be a diagram before Table 3-8 displaying the relationship between the objects:
Actual definition, check definition, check link and Parameter definition.
They are connected the following way:
CheckLink.source = ParameterDefinition (3.5.3.6)
CheckLink.related = CheckDefinition (3.5.3.5)
CheckDefinition.source indicates which actual definition object it uses. (3.5.3.3)

An example is provided in “Figure 2-4: Activity Chaining Example” of the COM book, page 22 (521.1-B-1)

CATEGORY OF REQUESTED CHANGE:
Technical Fact ___ Recommended ___ X ___ Editorial ___

NOTES:

SUPPORTING ANALYSIS:
It would be easier for the reader to understand the object connections and it would remove the need to draw or scribble them on a separate paper.

DISPOSITION:
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

3.4.9 RID 10

REVIEW ITEM DISPOSITION (RID):
RED BOOK RID INITIATION FORM

AGENCY RID NUMBER: CC_10
SUBMITTING ORGANIZATION (Agency, Center): ESA

REVIEWER'S NAME: César Coelho
CODE:
E-MAIL ADDRESS: Cesar.Coelho@esa.int
TELEPHONE:

DOCUMENT NUMBER: Red Book, -
DOCUMENT NAME: All MO books
DATE ISSUED: -
PAGE NUMBER: -
PARAGRAPH NUMBER: -
RID SHORT TITLE: Page numbering

DESCRIPTION OF REQUESTED CHANGE: (Use From: "..." To "...", format)
The page number of the pdf should match the written page number.

CATEGORY OF REQUESTED CHANGE:
Technical Fact ___ Recommended ___ X___ Editorial ___

NOTES:
TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)
RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.
EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

SUPPORTING ANALYSIS:
The page numbers should not be divided by sections. Nowadays, it is much more important that they match the pdf page number.
For example, a conversation on the phone becomes more difficult when referencing a certain page number if one side has a hard-copy and the other side has a digital version.
This type of mistake is acceptable for old books but it shouldn’t happen in any book made in the 21st century.

DISPOSITION:
3.4.10 **RID 12**

**REVIEW ITEM DISPOSITION (RID):**  
**RED BOOK RID INITIATION FORM**

**AGENCY RID NUMBER:** CC_12  
**SUBMITTING ORGANIZATION (Agency, Center):** ESA

**REVIEWER'S NAME:** César Coelho  
**CODE:**  
**E-MAIL ADDRESS:** Cesar.Coelho@esa.int  
**TELEPHONE:** -

**DOCUMENT NUMBER:** CCSDS 522.1-R-3  
**DOCUMENT NAME:** MO Monitor and Control Services  
**DATE ISSUED:** March 2014  
**PARAGRAPH NUMBER:** -

**RID SHORT TITLE:** Monitoring List concept does not exist

**DESCRIPTION OF REQUESTED CHANGE:** (Use From: "..." To ",...", format)

**PUS, PUS-C compliance**  
Add a CheckMonitoringList object with four fields: name (type: MAL::Identifier), description (type: MAL::String), listEnabled (type: MAL::Boolean), and functionalMonitoring (type: FunctionalMonitoring).  
The FunctionalMonitoring object is explained in RID CC_13.  
For the functionalMonitoring field, the comment shall be: "If NULL, then the object represents a Monitoring List, else it represents a Functional Monitoring List"  
The advantage of the CheckMonitoringList (as defined above) is that it can represent a simple Monitoring List (PUS) or a Functional Monitoring List (PUS-C) based on their functionalMonitoring field Nullability.  
These objects can be enabled or disabled based on their listEnabled field.

**CATEGORY OF REQUESTED CHANGE:**

Technical Fact **X**  
Recommended ___  
Editorial ___

**NOTES:**

**TECHNICAL FACT:** Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)

**RECOMMENDED:** Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.

**EDITORIAL:** Typographical or other factual error needing correction.  
(This type of change will be made without feedback to submitter.)

**SUPPORTING ANALYSIS:**

In PUS a check can be enable/disable at both the parameter level and at the check definition level.  
Currently, M&C is only enabled/disable at the check definition level.

In PUS, page 119: “...the service maintains a monitoring list and checks parameter samples according to the information contained therein.”

page 120: “...which determine: a. Whether the monitoring of parameters is enabled or disabled at service-level.”

In PUS-C (PUS_C Draft 8[20140422]): “Functional Monitoring subservice – The functional monitoring subservice provides the capability to monitor the functional health of on-board elements (…). A functional monitoring definition includes a set of one or more parameter monitoring definitions: when a minimum number of these definitions is contemporaneously violated, that functional monitoring definition is considered violated and the associated report is issued.”
3.4.11 RID 13

REVIEW ITEM DISPOSITION (RID):
RED BOOK RID INITIATION FORM

AGENCY RID NUMBER: CC_13
SUBMITTING ORGANIZATION (Agency, Center): ESA

REVIEWER'S NAME: César Coelho
CODE: -
E-MAIL ADDRESS: Cesar.Coelho@esa.int
TELEPHONE: -

DOCUMENT NUMBER: CCSDS 522.1-R-3
DOCUMENT NAME: MO Monitor and Control Services
DATE ISSUED: March 2014
PARAGRAPH NUMBER: -

DESCRIPTION OF REQUESTED CHANGE: (Use From: "...") To "...") format)
PUS-C compliance

Add a FunctionalMonitoring object with four fields: monitoringList (type: List<MAL::Long>), protected (type: MAL::Boolean), minimumFailures (type: List<MAL::UInteger>), and condition (type: ParameterExpression).

For the monitoringList field, the comment shall be: "The list shall contain the object instance identifiers of the check definitions associated with the functional monitoring list".

It would also be necessary to create 6 new operations to manage the Functional Monitoring List: listMonitoringList, addMonitoringList, removeMonitoringList, enableMonitoringList, protectMonitoringList, and getMonitoringListReportStatus.

CATEGORY OF REQUESTED CHANGE: Technical Fact ___X__ Recommended ___ Editorial ___

NOTES:
TECHNICAL FACT: Major technical change of sufficient magnitude as to render the Recommendation inaccurate and unacceptable if not corrected. (Supporting analysis/rationale is essential.)
RECOMMENDED: Change of a nature that would, if incorporated, produce a marked improvement in document quality and acceptance.
EDITORIAL: Typographical or other factual error needing correction. (This type of change will be made without feedback to submitter.)

SUPPORTING ANALYSIS:
In PUS-C (PUS_C Draft 8[20140422]): “Functional Monitoring subservice – The functional monitoring subservice provides the capability to monitor the functional health of on-board elements (...). A functional monitoring definition includes a set of one or more parameter monitoring definitions: when a minimum number of these definitions is contemporaneously violated, that functional monitoring definition is considered violated and the associated report is issued.”.

DISPOSITION:
3.4.12 **RIDs feedback**

The first RIDs were submitted on the 14th of May 2014 directly to Dr. Mario Merri, the chairman of the Spacecraft Monitor & Control working group (SM&C). Later, on the 2nd of June 2014, two other RIDs (RID 12 and RID 13) were resubmitted to substitute RID 03 and RID 11 since they were not complete and/or not entirely accurate on their content.

On the 7th of July 2014, Sam Cooper sent the replies via email regarding the submitted RIDs.

**RID 01**

RID Disposition: Partially Accepted  
RID Response: “This seems to be a RID on the COM specification due to the structures specified being COM structures. However I see the issue and will provide a resolution in this specification that should be back ported into COM when appropriate.”

**RID 02**

RID Disposition: Rejected  
RID Response: “This would be provided by the Common::Configuration service however it would not explicitly be defined for Aggregation.”

**RID 04**

RID Disposition: Accepted

**RID 05**

RID Disposition: Partially Accepted  
RID Response: “This should not be possible because updateDefinition should not be used to make such a modification. However, as it stands this could be seen as a possibility. I shall clarify Alert::updateDefinition (and all other updateDefinition operations) what can and cannot be changed using updateDefinition.”

**RID 06**

RID Disposition: Accepted

**RID 07**

RID Disposition: Accepted

**RID 08**

RID Disposition: Rejected  
RID Response: “This is an implementation aspect for MO.”
RID 09
RID Disposition: Accepted
RID Response: “I will provide an object relationship diagram for each service.”

RID 10
RID Disposition: Rejected
RID Response: “This is not something that I can alter as I think it is a CCSDS norm. However this could be raised internally in CCSDS going forward.”

RID 12
RID Disposition: Duplicate CNES-BB_63 (Later: Accepted)
RID Response (CNES-BB_63): To be discussed.

RID 13
RID Disposition: Rejected (Later: Partially Accepted)
RID Response: “This adds more complexity to an already complex area and it is something that can be implemented in other ways.”

During the discussion, it was agreed that the Functional Monitoring function would be included on the Check service and a work around for the enabling/disabling of the PUS monitoring List would be provided by other means.

The agreed object structure follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MAL::Identifier</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>MAL::String</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>enabled</td>
<td>MAL::Boolean</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>monitoringList</td>
<td>List<a href="">MAL::Long</a></td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>protected</td>
<td>MAL::Boolean</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>minimumFailures</td>
<td>MAL::UInteger</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>condition</td>
<td>ParameterExpression</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>parameterId</td>
<td>COM::ObjectKey</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>operator</td>
<td>COM::ExpressionOperator</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>useConverted</td>
<td>MAL::Boolean</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>Mal::Attribute</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
4 IMPLEMENTATION OF 2 CCSDS MO SERVICES ON A MITYARM BOARD

4.1 Introduction

The main idea is to implement two of the CCSDS MO services, respectively, the Aggregation service and the Parameter service, on a MityARM-5CSX processor card operating as the Service Provider. The deployment of the software demonstrates the practicability of both the Aggregation service and the Parameter service, and tests the principles of the Mission Operation Service Concept using the new CCSDS Standards. As part of a larger system, the provider transmits the information to the consumer using the Message Abstraction Layer (MAL), which unambiguously defines a set of rules governing the syntax, semantics, and synchronization of communication between entities as a standard abstract API for transports. [2]

On the Service Consumer side, the message shall be decoded and the implementation of a MO/PUS adapter shall provide readable Packet Utilization Standard (PUS) packets which shall be interpreted by the Mission Control System (MCS) software infrastructure SCOS-2000 developed and currently maintained by the European Space Agency. [7]

![Diagram of Consumer-Provider linkage]

The project was originally designed to be developed without a consumer side but in order to test all the implemented operations, a Graphical User Interface consumer was developed as well.

4.2 MITYARM-5CSX and OPS-SAT relation

The MityARM-5CSX is a highly configurable System-on-Module that features an Altera Cyclone V (Dual-core Cortex-A9 32-bit RISC processor) with a 800 MHz max clock speed which is capable of running real-time operating systems. The module also includes a NOR FLASH and 1GB DDR3 RAM memory subsystems. [8] OPS-SAT will be a satellite devoted to demonstrate the improvement in capabilities which will arise from the use of more powerful on-board computers. OPS-SAT will use the MityARM-5CSX. [9]

---

[8] OPS-SAT mission: http://www.esa.int/Our_Activities/Operations/OPS-SAT
4.3 Software Architecture: General Overview

A general overview of the software architecture is shown on the following picture.

![Software Architecture - General Overview](image)

---

10 Critical Link website: [MitySOM-5CSX Overview](#)
4.4 Provider side - CLI

The provider does not have a graphical user interface because it is intended to be used without one. The application was developed in Java using NetBeans.\[11\]

The following picture shows the command-line interface (CLI) of the provider application after start-up and ready to be used.

![Figure 7: Provider running and ready for operation](image)

4.4.1 Parameter Service

Parameter status monitoring is performed by publishing the status of a set of predefined monitoring parameters that contain status information. Monitoring parameters have an evolving status represented by a chronological sequence of status updates which can be periodic, change-based, or a mixture of the two. The service consumer will be able to: subscribe for parameters, receive updates, modify requested parameter list, unsubscribe.\[12\]

\[11\] NetBeans: [Official Website]

The M&C parameter service does not define the provided parameter information in the service specification; it delegates the definition of the provided parameter information to the runtime configuration of the provider by defining what the structures look like, and how the parameter information is returned. Parameter definitions are managed using the operations of the Parameter service.

### 4.4.2 Aggregation Service

The Aggregation Service provides the capability to acquire several parameter values in a single request. The aggregation might be predefined by the service provider (e.g. housekeeping parameters) or predefined at runtime by the consumer (e.g. diagnostic report). The user can, for example, request a data product that includes the GPS position longitude and latitude of the spacecraft. This can be achieved by reading the appropriate GPS instrument on-board (simulated in this case) and returning the data as a set. The aggregation service provides operations to define which parameters to aggregate and reports the current values of those parameters.

### 4.4.3 Parameter Manager

Holds the data from the Parameter Service such as the Parameter Definitions, the respective object instance identifiers and some other necessary management variables. Provides also operations for the management of the Parameter Definitions: add, update and delete. Storing and Loading the data from the hard-drive is also taken care in this block. It has the secondary function of interacting with the Monitoring Parameters layer in order to acquire the respective values of each Parameter Value.

### 4.4.4 Aggregation Manager

Holds the data from the Aggregation Service such as the Aggregation Definitions, the respective object instance identifiers and some other necessary management variables.

### 4.4.5 Monitoring Parameters

Provides a middle layer between the Simulated Instruments and the Parameter Manager. The only function of this block is to return a parameter value for a certain reserved string name. This block was designed to allow the evolution of the project, for instance, allow the replacement of the Simulated Instruments block by real instruments without modifying any of the blocks above it on Figure 6: Software Architecture - General Overview.

### 4.4.6 Instruments Simulator
Fine ADCS: In this block only a Magnetometer was simulated. The value of the Magnetic field is computed using the dipole model of Earth's magnetic field. [13]

GPS: In this block a GPS is simulated. The block uses the current time of the machine and computes the position of a spacecraft in a dawn-dusk orbit at an altitude of around 650 kilometers. Random errors were purposely introduced in the model to have small variations on the returned value.

4.5 Consumer side - GUI

The consumer application has different tabs for the interaction with the provider. The Home tab is the default tab after start-up and allows the user to configure the rmi links, to connect to the provider and contains some general information regarding the application operation.

The following picture shows the graphical user interface (GUI) of the consumer application after start-up and ready to be used.

![Consumer Interface (CCSDS MO)](image)

**Figure 8:** Consumer application Home tab after start-up

---

[13] Dipole model of the Earth's magnetic field
Comparison of the CCSDS Mission Operations services with the Packet Utilization Standard services

Figure 9: Parameter Service Tab of the Consumer GUI

Figure 10: Published Parameter Values Tab of the Consumer GUI
4.6 Mity USB Terminal

This application was developed during the internship and displays the information on the CLI of the board via a USB cable connection. The application was coded in C++ on the QT platform.
5 CONCLUSIONS

The Master Thesis resulted in several contributions for the enhancement of the CCSDS MO services and provided many direct results. First, the comparison summary which covers the main advantages and disadvantages between PUS/PUS-C services and CCSDS Mission Operations services (particularly the Monitor and Control services). Second, the systematic comparison process used, originated a very detailed technical document which contains all the PUS and PUS-C (up-to-date) telemetry and telecommands packets and match them with their equivalent operations and objects from the CCSDS MO standard.

From the obtained comparison summary, the advantages and disadvantages were transparently identified, and this allowed the submission of 11 RIDs in order to fix the problems found and also some enhancement suggestions were proposed on the CCSDS MO M&C book. The outcome of the RIDs came back in July and the majority of the suggested solutions were accepted or partially accepted. These include: The possibility to represent an unknown number of execution stages; The possibility to enable the CCSDS MO M&C Check service at the service level; A change in requirement 3.5.2.33 (related with the optionality of the maxReportingInterval); New requirements to limit the updateDefinition operation on modifying the Severity field of the AlertDefinition object in order to avoid chronological errors; Some editorial typos and mistakes shall be fixed; A new operation named setParameterValue in the Parameter Service shall be introduced; An object relationship diagram for the Check Service; A new “Functional Monitoring” equivalent PUS-C subservice.

The RIDs that were not accepted either will be implemented in the future or helped raising internal awareness of their existence in the Spacecraft Monitor and Control working group.

The second project, the Implementation of two CCSDS MO services: Parameter service and Aggregation service on a MityARM board, resulted in many other questions and suggestions directly submitted via email to Sam Cooper (no RIDs were raised). These included: The modification of the AggregationValue structure to match how the AggregationDefinition stores its parameters which is a list of parameters within a list of parameterSets; Some requirement suggestions were also sent regarding the Aggregation service: The recognition that the word “timestamp” was not always being written the same way throughout the book (“time-stamp” and “time stamp” were also being used to refer to the exact same concept).

It is important to mention that this project was not originally part of the Master Thesis but only an optional task, although, as shown from the above paragraph, it resulted in a set of contributions which also allowed the enhancement of the CCSDS MO Standard.

The systematic mapping initialized during the internship at ESA can be updated once the next CCSDS Mission Operations Monitor and Control draft 4 is released.

The second project, the implementation of 2 CCSSDS MO services, can also be expanded by including the remaining services from CCSDS MO M&C and on the near future, updated with the new modifications that will be introduced by the next CCSDS MO M&C draft 4 version.
6 ACKNOWLEDGEMENTS

First and foremost, I would like to begin by thanking my supervisor Dr Mehran Sarkarati for giving me the opportunity to develop my Master Thesis at the European Space Agency on the Applications and Special Projects Data Systems Section in Darmstadt and for providing me the necessary help whenever I needed during my internship.

Second, I would like to thank my family for giving me the necessary support and motivation for living in a foreign country without the need to worry about security or safety needs. This allowed me to focus entirely on my self-actualization and personal development.

I would like to express my deep gratitude to Sam Cooper which helped me a considerable number of times through numerous exchanged emails with questions related with CCSDS Mission Operations Monitor and Control. I would also like to thank Brigitte Behal which took the time to review my comparison summary, Serge Valera for providing me help with question regarding PUS-C, Jochen Klein which helped me fixing a GPS simulation problem, and also, Dr. Mario Merri for the encouragement given during the whole internship.

I would like to thank Professor Peter von Ballmoos for being very helpful throughout the last academic year, not only to me but also to all my other colleagues and for being my university supervisor from the Université Paul Sabatier. I would also like to thank Professor Johnny Ejemalm for following my progress and for taking his time to review my Master Thesis as supervisor from Luleå University of Technology.

Finally, I would like to thank my friend Nader Samir not only for being always very supportive but also for his excellent collaboration in many university projects where we worked together throughout the Space Master program. Marina Gruet also deserves a mention in this thesis for helping me with French related matters and for her constant friendly support.