Research on Modeling of Satellite On-orbit Operation Procedure

LV Wei, HE Xiaoyu, QIN Wei
Beijing Institute of Spacecraft System Engineering, Beijing 100094, China
lw_bit@163.com

Abstract: Aiming at the demand of the TT&C system for the on-orbit satellite to automatically perform various operations, the paper proposes a modeling technology for on-orbit operation procedures. Based on the formal method, a grammer specification of operating rules is formulated, and modeling rules of operating rules are defined by XML Schema. The process of graphical editing, serialization and deserialization of operating rules is presented. Practical application shows that it can improve the design and execution efficiency of operating procedures.

1. Introduction
During the on-orbit lifecycle of satellites, it is necessary to use the TT&C system[1] to arrange the satellite operation according to certain logic and form control instructions. Through the station, instructions are injected into the satellite, and the execution effect of the instructions is interpreted, so as to complete the TT&C operation, payload operation and abnormal handling of the satellite. With the increasing number of satellites on orbit and the complexity of satellite functions, requirements for the TT&C system are increasing[2]. Efficient assignments for on-orbit operations, generalization and automatic execution of the operations, has become the core issue to be solved for the TT&C system.

2. Procedure Language Grammar
For all kinds of operations, standard and automatic procedures need to be designed. To describe the operation procedures of on-orbit satellites, we need a fully functional language[3] which covers all the atomic operations and logical structures of on-orbit operations. Through the analysis and induction of the on-orbit operation process, based on normal form rules in BNF[4], the grammar of the operation rules if formally defined as follows.

2.1. Basic Paradigm

DIGIT::= “0”| “1”| “2”| “3”| “4”| “5”| “6”| “7”| “8”| “9”
DEC_NUMBER ::= <DIGIT> | <DIGIT> <DEC_NUMBER>
HEX_CHAR ::= <DIGIT> | “A”| “B”| “C”| “D”| “E”| “F”
HEX_NUMBER ::= <HEX_CHAR> | <HEX_CHAR> <HEX_NUMBER>
NON_DIGIT ::= “A”| “B”| “C”| “D”| “E”| “F”| “G”| “H”| “I”| “J”| “K”| “L”| “M”| “N”| “O”| “P”| “Q”| “R”| “S”| “T”| “U”| “V”| “W”| “X”| “Y”| “Z”
CHART ::= <NON_DIGIT> | <DIGIT> | “ ”
TEXT ::= <CHART> { <CHART> }
represents non-numeric character, CHART represents character, and TEXT represents string.

2.2. Telecommands Paradigm
Send telecommands to the satellite and set additional parameters of the telecommands. The paradigm is as follows:
\[
\text{SEND\_STATEMENT} ::= \text{“send”}\ \text{“(”}\ <\text{TC\_NO}\ [\text{“,”}\ <\text{PARAM\_LIST}\ ]\ “)}
\]
TC\_NO ::=<NON\_DIGIT> {NON\_DIGIT}{DIGIT}
PARAM\_LIST ::= <HEX> [“,” <HEX\_NUMBER >]
TC\_NO represents telecommand code. PARAM\_LIST represents additional parameters of the telecommands.

2.3. Telemetry Interpretation Paradigm
According to the preset interpretation expression, the real-time data of the telemetry parameters can be interpreted as expected. The paradigm is as follows:
\[
\text{VERIFY\_STATEMENT} ::= \text{“verify”}\ \text{“(”}\ <\text{EXPRESSION}\ \text{“,”}\ <\text{DEC\_NUMBER}\ “)}
\]
EXPRESSION ::= <TM\_NO> <relational\_operator> (<DEC\_NUMBER>|<TM\_NO>) {<logical\_operator> <VERIFY\_STATEMENT>}
TM\_NO ::=<NON\_DIGIT> {NON\_DIGIT}{DIGIT}
relational\_operator ::= “>” | “>=” | “<” | “<=” | “==” | “!=”
logical\_operator ::= “and” | “or” | “not”
TM\_NO represents telemetry parameter code. “relational\_operator” represents relational operator. “logical\_operator” represents logical operator.

2.4. Wait Paradigm
Block and wait for certain seconds. The paradigm is as follows:
\[
\text{WAIT\_STATEMENT} ::= \text{“wait”}\ \text{“(”}\ <\text{DEC\_NUMBER}\ “)}
\]
The value following the key word “wait” represents the waiting time (unit: “second”).

2.5. Manual Confirm Paradigm
Suspend procedure execution and show tip for the operator to confirm. The paradigm is as follows:
\[
\text{ASK\_STATEMENT} ::= \text{“ask”}\ \text{“(”}\ <\text{TEXT}\ “)}
\]
Texts following the key word “ask” represent information to be confirmed by the operator.

2.6. Static Caption Paradigm
With no action but show a plain caption. The paradigm is as follows:
\[
\text{LABEL\_STATEMENT} ::= \text{“label”}\ \text{“(”}\ <\text{TEXT}\ “)}
\]
Texts following the key word “label” represent the caption.

2.7. Abort Paradigm
Abort execution of the procedure. The paradigm is as follows:
\[
\text{ABORT\_STATEMENT} ::= \text{“abort”}
\]

2.8. Logic Paradigm
Implement branch or loop logic. The paradigm is as follows:
\[
\text{LOGICAL\_STATEMENT} ::= (“if” | “for”)\ \text{“(”}\ <\text{EXPRESSION}\ “)}\ \text{“{”}\ <\text{STATEMENT}\ “)}\]
\[
\text{SINGLE\_STATEMENT::=}<\text{SEND\_STATEMENT}|<\text{VERIFY\_STATEMENT}|<\text{WAIT\_STATEMENT}|<\text{ASK\_STATEMENT}|<\text{LABEL\_STATEMENT}|<\text{ABORT\_STATEMENT}\]
\[
\text{STATEMENTS ::=}<\text{SINGLE\_STATEMENT}> | <\text{SINGLE\_STATEMENT}> “ ” <\text{STATEMENTS}>\]

3. Procedure Modelling, Serialization and Deserialization
Based on the formal definition of the procedure grammar, and according to demand of on-orbit
operation, operation program can be designed\(^5\). Traditionally, satellite operators write operation program scripts directly and then compile and run the program by compiler\(^6\). The paper proposes a graphical modeling method for the design of operation procedures, which is more efficient and intuitive than writing scripts or tabular documents\(^7\)\(^8\), and has good readability and maintainability.

### 3.1. Procedure Model

The paper uses XML Schema to establish the entity model of operating procedures. The model is based on the formal syntax of operating procedures, and offers an XML representation for the procedures so that the operating procedures can be saved as an XML file and loaded from the XML file to graphical interface. The XML definition is shown in the following figure.

![Fig.1 Automatic Procedure Model Structure](image)

The model root element is “Procedure”, which represents the operating procedure. It contains 0 to multiple (minOccurs=0, maxOccurs=unbounded) sequence of “ProcedureStep” type elements.

The type “ProcedureStep” is a complex type that represents a single step in a sequence of procedures. “ProcedureStep” has attribute and sequence of child elements. The attributes of “ProcedureStep” are show in table 1. “ProcedureStep” type’s sub element sequences are also “ProcedureStep” type, thus supporting multilayer nesting of procedures.

| Attribute   | Description             | Type     | Use     | Constraint          |
|-------------|-------------------------|----------|---------|---------------------|
| stepId      | Step number             | string   | required| None                |
| executionTime| Absolute time to execute| datetime | optional| None                |
| keyword     | Operation keyword       | simpleType | required| None                |
| event       | Operation description   | string   | optional| None                |
| tc          | Telecommand code        | string   | optional| None                |
| tm          | Telemetry parameter criterion expression | string | optional | None                |
| data        | Additional parameters of Telecommand | string | optional | None                |
| timeout     | Time out for the step itself | string | optional | None                |

### 3.2. Editing, Serialization and Deserialization

Since we have an XML Schema definition for the procedures, next we can edit, store, and load the procedures through a tabulated graphical user interface. The detailed process is show in the following figure (with the Java language implementation as an example).
Fig.2 Procedure model graphical editing, serialization and deserialization

1) Through the JAXB tool, the XML Schema of the operating procedure is mapped to the Java beans of the operating procedure;
2) Make a data binding between graphical table and the Java Bean arrays of the procedure;
3) According to the requirements of on-orbit satellite operations, add new step lines and edit attributes of each step line until all the steps of the procedure finishing;
4) By using data binding, transfer the procedure in the graphical table to Java Bean arrays;
5) Through the JAXB tool, serialize the procedure’s java bean arrays to an XML file, which conforms to the procedure’s XML Schema constraints established in step 1;
6) With the JAXB tool, we can deserializate the XML files of procedures into Java Bean arrays. By using data binding, procedures can be displayed and edit in the graphical table editor.

By this process, we complete the formalization definition, XML Schema definition, graphical design, serialization, preservation, deserialization, loading display and other key modeling issues of on-orbit operation procedures.

4. Application Effect
Based on the above research results, the satellite on-orbit operating software can realize the design of operating procedures and automatic execution. The system has been deployed at the ground station and applied to the on-orbit operation of multiple communication satellites, realizing operations like position keeping, gyro calibration. The system interface is as follows.

Fig. 3 User interface of procedure design

5. Conclusion
The paper focuses on the description and modeling of satellite on-orbit operating procedures, and gradually conducts research from abstract to specific. Through the formalization, the XML Schema...
definition, graphical display and editing, saving (serialization) and loading (deserialization), we complete the operating procedure modeling process. The on-orbit operation software developed based on the research results can cover various types of on-orbit satellite operation requirements, realize the operating procedures of graphical design and persistent storage. It is easy to load, display and execute. The established operating procedures have good practicability, readability, reusability and maintainability, and have a good practical application effect on improving the management and automatic operation capability of satellite on-orbit operating procedures.

References
[1] Guan Hui, Ning Yongzhong. The Development Course and an Outlook on Chinese Satellite TT&C[J]. Space International. 2018, (1):55-59
[2] XU Aimin, ZHANG Guoting. Discussions on TT&C Management of Commercial Space in China[J]. Journal of Spacecraft TT & C Technology. 2017, 36(3):157-163
[3] Murthy T K, Munch R E. Computer Application in Spacecraft Design and Operation[M]. New York: Springer-Verlay Incorporated, 2003.
[4] Sun Bo, Ma Shilong, Yu Dan. Spacecraft automatic test and spacecraft test language[J]. JOURNAL OF BEIJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. 2009, 35(11):1376-1378
[5] Britton K J, Schaible D M. Spacecraft testing programs: adding value to the systems engineering process[Z]. Kennedy space center; Langley Research Center, 2011.
[6] HE Yaoqiang, YANG Renbao, ZHANG Wuguang. Design and Implementation of High-Level Language for Spacecraft Control[J]. Journal of Spacecraft TT & C Technology. 2012, 31(3): 62-65
[7] Zhu Weibao, Wang Jinrong, Li Diqing. Spacecraft Automatic Testing Model and Automatic Testing Language Design[C]. The 19th National academic annual meeting of TT & C, measurement and instruments. 2009(s):118-121
[8] Li Zhengyu, Liu Jian, Li Xuzhi. Design and Implementation of a New Satellite Testing and Operation Tool Software[J]. COMPUTER ENGINEERING AND APPLICATIONS. 2005, 41(4):116-118