Test Generation for VMF Tactical Data Link Messages - Coping with Message Length Variability and Semantic Message Rules

Jihyun Lee[1], Sungwon Kang[1], Myungchul Kim[1], Changsup Keum[1], Kyungmin Go[2], Danhyung Lee[1]

[1]*Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, Korea*
{jhyunlee, sungwon.kang, mck, cskeum, danlee}@kaist.ac.kr
[2]*LIGNex1, 838 Yeoksam-dong, Gangnam-gu, Seoul, Korea*
kyungmingo@lignex1.com

***Abstract*—As modern militaries are evolving their military command and control systems, it has become an important issue to test tactical data systems (TDSs) using Variable Message Format (VMF) messages because VMF is a common means for exchanging tactical data between combat units at various organizational levels. VMF has presence/recurrence fields allowing the user to send a minimal length message without losing information, but it causes the length of a message to vary, thereby making message validation non-trivial. Furthermore, VMF includes semantic message rules that help define precisely the interrelationship between the values of different fields, but it makes validating such relationships difficult. This paper presents our method and evaluation results for overcoming these challenges by applying a systematic test generation method for VMF Tactical Data Link messages. Our method consists of the principles and the derivation of the test requirements, as well as a procedure for test generation based on them. We applied the method in developing a VMF test tool and used the tool to test the various versions of a VMF message parser. We confirmed our method was effective in reducing the complexity of testing that arises due to the message length variability and semantic rules.**

1. Introduction

The Variable Message Format (VMF) standard[1] was developed by the Department of Defense to specify the formats of variable length messages for tactical data links (TDLs). Flexibility of messages provided by message length variability and precise defining mechanisms exhibited by semantic message rules for VMF messages are considered essential for the continued operation of automated tactical data systems (TDSs), which must exchange mission critical command and control information in a timely manner. VMF is a bit-oriented digital information standard having variable message length that allows the user to send only the required information by using presence/recurrence fields[7]. For VMF interoperability between Services, Agencies (S/As) and TDSs, the following VMF related components are needed: a VMF Integrated Database (VID) that contains all the up-to-date information of the evolving VMF standard, a Real-time Parser Generator for rapid incorporation of changes in the VMF standard, and Gateway Software for exchanging tactical information between different tactical data links and with legacy systems. For testing and verification of standard conformance or interoperability[8] of the VMF message protocol implementations, the development of a VMF test tool is needed as well.

The VMF test tool must validate not only the TDSs' capability of correctly interpreting VMF messages as specified in the VMF standard but also its ability to report the types of errors that a VMF message contains in the case of receiving an invalid message. The VMF test tool needs to detect missing mandatory fields, determine which groups have exceeded the maximum number of repetitions, validate cases and conditions, and report any field errors. However, because there are already more than 100 operating VMF messages and new VMF messages will be added whenever necessary, it is important to ensure that the VMF test tool can adequately validate the existing VMF processing implementations. Therefore, an automatic test generation for VMF testing is highly desired.

Generating VMF test messages poses two major challenges. The first challenge is that the length of a VMF message is variable because VMF has presence/recurrence fields for exchanging only the required information. The second challenge is that a VMF message includes semantic message rules (or semantic rules, in short) for defining relationships between different fields as well as for defining non-syntactical constraints for individual fields. For automated conformance testing of the Link 16 protocol, which is another tactical data link protocol, E. Mak et al. developed a technique that uses Discrete Event System Specification (DEVS) and XML[10], but the main focus of the technique was testing the behavioral aspects of Link16. For VMF message testing, the VMF Test Tool (VTT) [3] is being used in U.S. Army but its test generation approach for testing VMF standard conformance is not known. To date there have been no known studies related to VMF test generation. Therefore, it is necessary to develop a systematic test generation method for ensuring reliability of evolving VMF messages and new protocols using variable length messages.

In this paper, we first present the principles that we adopted for meeting the two challenges mentioned above. To meet the first challenge, we adopt message segmentation and slicing principles. To meet the second challenge, we use Disjunctive Normal Form (DNF) for

formally specifying VMF messages and two-pass test generation principles. Then we define the test requirements based on the results of analyzing the VMF standard and a procedure for VMF message validation. As the general area of protocol testing is to test whether a protocol implementation conforms to its specification[11, 13] it should test both protocol behavior and protocol messages. This paper only deals with test generation for VMF messages and a tool for automating it. However, our method is a general method for message testing and therefore it can be used with other protocol testing methods[12, 14, 15, 16] for their message testing part.

This paper is organized as follows: Section 2 describes VMF messages for background knowledge. In Section 3, we explain test generation principles that we used to meet the challenges of generating VMF test messages. Section 4 describes test requirements of VMF messages that are derived for defining what aspects of VMF messages should be tested. Section 5 describes our test generation method which consists of principles, test requirements, and a procedure. Section 6 explains the application of the presented method in developing our VMF test tool and its evaluation results. Finally, in Section 7, we conclude our paper and suggest future directions for extending our work.

2. VMF Messages

The VMF standard was developed by the U.S. Army Communication and Electronic Command (CECOM) to support digital message exchange in the battlefield and to provide interoperability among combat units. VMF messages are being used as a common means for exchanging tactical data between combat units at various organizational levels. A VMF message is bit oriented and designed to let the user send only the required information using presence/recurrence fields and this also serves the need of real time digital information exchange in bandwidth-constrained environments.

A VMF message consists of a Data Field Identifier (DFI), Data Use Identifier (DUI), and Data Item (DI). Information such as position/location data, navigation data, etc. contained in a VMF message in the form of a bit stream needs to obey the syntactic and semantic rules specified in the VMF standard. Fig. 1 is an example of a VMF message specification.

INDEX NO.	REFERENCE DFI/DUI	DUI NAME	# BITS	CAT	GROUP CODE	REPEAT CODE
1.1	4045001	GRI	1	M	R1	(64)
1.2	4004012	URN	24	M	R1	
	...					
1.5	4119002	LOC DERIV.	24	M	R1	
1.6	4014002	FPI	1	M	R1	
1.6.1	4119005	LOC. QUA.	4	X	R1	
…						

Figure 1. Specification of a VMF message (K05.1)

The syntactic field, called the *indicator*, consists of a single bit and indicates whether there is specific data immediately following it. There are two kinds of indicators in VMF messages; presence and recurrence indicators. A presence indicator consists of a Field Presence Indicator (FPI), which indicates whether an information field exists, and a Group Presence Indicator (GPI), which indicates whether a group of fields exists. The Field Recurrence Indicator (FRI) and Group Recurrence Indicator (FRI) indicates whether an information field or a group of fields is repeated, respectively. The indicator value '1' indicates that the relevant field or the relevant group of fields exists for a presence indicator, and it indicates that it is repeated for a recurrence indicator. The indicator value '0' is similar except that it indicates that the relevant field or the relevant group of fields does not exist and is not repeated. The length of a VMF message can vary depending on the existence or non-existence of information fields as indicated by such indicator fields.

Semantic rules for defining relationships between different fields are described in the form of CASE statements or CONDITION statements. The CASE statement is similar to the switch-case statement of the C language. The CONDITION statement that represents a constraint between fields is similar to the IF-THEN-ELSE statement of the C language. Fig. 2 is an example of a CASE statement.

```
Case: Operations Plan or Operations Order
THIS CASE REQUIRES
        PLAN/ORDER TYPE [4093/007] is specified "0"
        (OPERATIONS PLAN)
XOR PLAN/ORDER TYPE [4093/007] is specified "1"
      (OPERATIONS ORDER)
AND FPI for OPERATION IDENTIFICATION [4003/007] is
          specified "1" (PRESENT)
...
AND GPI for G4 is specified "1" (PRESENT)
AND GPI for G7 is specified "1" (PRESENT)
END CASE
```

Figure 2. Example of a CASE statement

The characteristics of a VMF message can be summarized as follows: (1) The VMF message is bit stream and does not have white space between adjacent fields, (2) The VMF message length can vary as defined by field presence/recurrence indicators, (3) The VMF message must obey semantic rules as well as syntactic rules, and (4) Because the specification of a VMF message is stated in a natural language it can lead to ambiguous interpretation and much effort may be required for correct interpretation. Fig. 3 is an example of a VMF message. In the example, '|' is inserted as a separator of fields and does not constitute real content of a VMF message.

```
0|1|1|00000000000000010100000001100001110101010011|0|1|000110001
10110011001|10110000001010|1100100101000101l|01110101100000100
|0|1|00100010000011000101O|10010001000011|11100100100|001100|1
10100100001100l1|1|1|010|0|1|0010|1|0010|0|0|1|111100010000110
01|1|1|011000011110|110010000110011110001O|1|011|1|0011001|001
1|10010|01001|001000|000100|0|0|
```

Figure 3. VMF message example

The characteristics of a VMF message described above lead to several constraints from the test generation point of view. First, it is necessary to reduce the complexity of test generation before applying the existing test generation techniques, such as boundary value analysis and cause-effect graphing[17]. For this we decompose a VMF message into segments. Second, we need test messages that validate various complex semantic rules defining relationships between fields and messages. To that end we specify formally the semantic rules and perform two stages of test generation, i.e., generating test messages for semantic rules followed by generating messages for syntactic rules. Last, individual information fields can have range constraints. The specifications may put, for example, range constraints on individual information fields, and information fields related to time and date have intrinsic range constraints. The names of such information fields and their ranges are managed in the test database (TDB) and based on this information range error segments are generated.

3. Principles for Test Generation

In this section we present a set of test generation principles that we adopted to overcome the two major challenges mentioned in the Introduction. Principles 1 and 2 are new principles designed to meet those challenges. Principle 3 is an existing technique that has been used to minimize Boolean functions [9].

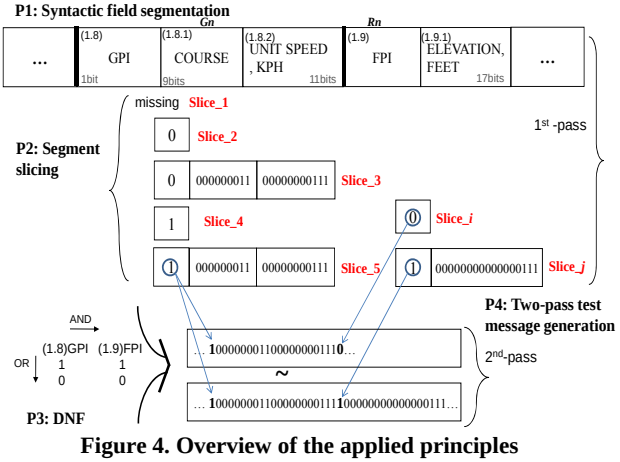

Figure 4. Overview of the applied principles

Figure 1 provides an overview that shows how the principles are applied to generate test messages. To meet the first challenge, we adopt message segmentation and slicing principles (P1 and P2). To meet the second challenge, we use Disjunctive Normal Form (DNF) for formally specifying VMF messages and two-pass test generation principles (P3 and P4).

3.1 Principles for meeting the message length variability challenge

The principles that we adopt to resolve length variability are message segmentation and slicing. A message is segmented on the basis of indicator fields to reduce the complexity of test generation. Each segment generates slices that consist of indicators' values, and the slices become the basis for test message generation. From a segment various slices are obtained as results of applying different test requirements on VMF messages. A test message is generated by combining such slices.

Principle 1 - Message segmentation

In the VMF message structure, a group is the biggest unit composing a message and a group indicator (GPI or GRI) provides information for grouping fields that belong to the same group. Since the group indicator indicates whether or not a sequence of fields follows, it provides an opportunity for reducing the complexity of test generation. Therefore, we divide a message into segments such that each group becomes one segment (Gn) and a sequence of fields that does not belong to any group also becomes one segment (Rn). For example, P1 in Fig. 4 and in Fig. 5 becomes a segment G1, because '1.8(GPI)', '1.8.1(COURSE)', and '1.8.2(UNIT SPEED, KPH)' make a group. . In a similar way, the second group indicator '1.11(GPI)' and its subsequent fields make a segment G2, and the fields that exist between G1 and G2 become a segment R2. Based on such segmentation, valid and invalid segments are produced as exemplified in Step 2 of our test generation procedure in Section 5. Segments produced in this way are stored in the test database (TDB).

Principle 2 – Segment slicing

For each segment we generate slices that consist of possible cases of indicator fields of the segment. A slice of a segment is either a valid bit stream or an invalid bit stream for that segment. A segment has two kinds of fields, i.e., indicator fields and information fields. When we generate message slices for a segment, we first consider GPI, FPI, and FRI indicator fields. With our basic test coverage strategy, at least two test messages should be generated for every indicator field and every information field with a range constraint. As shown in Fig. 4, in the case of an indicator field (GPI and FPI in Fig. 4) its value is either 0 or 1 and there will be one test message for each case. In the case of an information field

with a range constraint, there will be one test message for the case when the range constraint is satisfied and one for the case when it is not satisfied.

Next, information fields are considered. Information fields do not cause message length variability. However, because the VMF standard defines range values for specific information fields and there are also such fields as time, date, etc., which have predetermined ranges, range values should be considered separately when we analyze possible cases. Equivalence partitioning, boundary value analysis, and cause-effect graphing[6] can also be used for analyzing the possible cases.

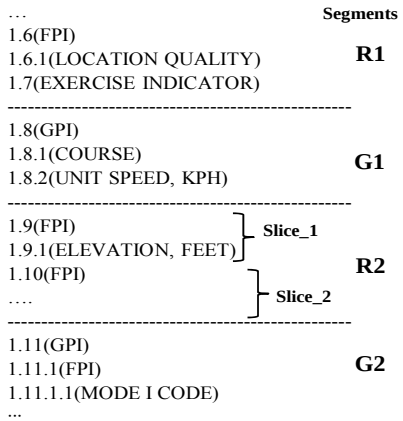

Figure 5. Message segmentation example (K05.1)

3.2 Principles for meeting the semantic rules challenge

An example of a semantic rule for a VMF message is shown in Fig. 3. This form is difficult to interpret and is also prone to mistranslation. To meet this challenge, we devised the Formal VMF (FVMF) language[4], as another component of our method, in order to formally specify VMF message formats, which are described with a natural language and a semi-formal language in the VMF standard. The FVMF uniformly represents diverse types of semantic rules in Disjunctive Normal Form (DNF)[9]. Then FVMF specification of VMF messages was used as the intermediate basis for developing our VMF test tool.[1] DNF has the advantage of simplifying semantic rules and constraints among fields because it is a normalized propositional logic expression in the form of a disjunction of conjunctive clauses.

In addition, since semantic rules can be validated only after the structure of the fields is known, it is efficient to utilize the segmentation results through Principles 1 and 2. Therefore, we adopt the principle of the two-pass test message generation, which requires that the message segments for validating semantic rules be generated after syntactic aspects have been considered.

Principle 3 - DNF

The third principle requires that semantic rules of VMF messages be transformed into DNF. Fig. 6 illustrates the procedure for transforming a CONDITION statement into DNF. The 'Condition', 'Then', and 'Else' statements of the CONDITION statement consist of AND, OR, and XOR operations between the values of various fields. First, a CONDITION statement is transformed into a Boolean expression that is composed of only 'AND' and 'OR' operations, which is then transformed into an equivalent DNF.

The example in Fig. 6 shows a transformation that applies the violation rule for transforming a Boolean expression of the semantic information of a VMF message into DNF. Since each row of a semantic rule table represents a violation of a semantic rule, if no rows of a table are satisfied (i.e., no rows evaluates to 'true'), a message is determined to be valid; otherwise it is deemed to be invalid. The advantage of this interpretation is that a violation can be confirmed uniformly within the application ranges of each row throughout the several DNF tables by defining each row of a table as an invalid relationship between fields.

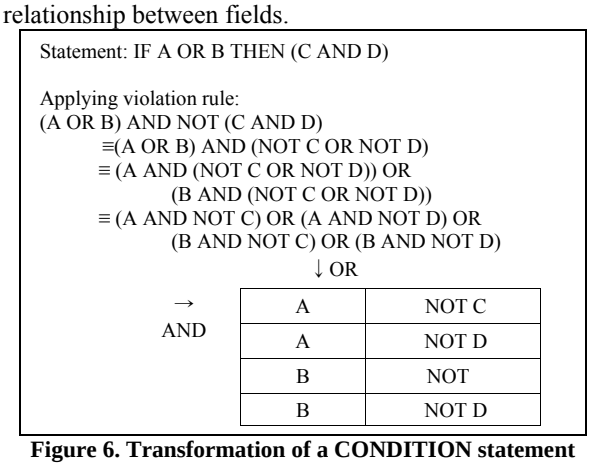

Figure 6. Transformation of a CONDITION statement

Fig. 7 describes the procedure for transforming a CASE expression of a VMF message into DNF. In Fig. 7, the two relevant fields are called A and B, respectively. Test messages for semantic rules are generated on the basis of DNF expressions, which are specified in FVMF, and decision tables (cf. Fig. 10) by applying a method such as the cause-effect graphing technique[6]. By retrieving the segments from the TDB, which have semantic rule-relevant fields on the decision table, test messages for semantic rules can be generated.

Principle 4 - Two-pass test message generation

[1] The associated tools, the VMF Parser Generator and the VMF Gateway Software, which were developed in other related subprojects within the tactical data link system development project, were also developed with the same FVMF specification of the VMF messages.

Since semantic rules are constraints among fields, the structure among fields should be known first in order for test generation to be performed. Therefore, test message generation must consider syntactic rules first and then semantic rules. During message segmentation, segments including slices related to semantic rules, have already been generated (1st-pass in Fig. 4). For generating semantic rule-relevant test messages, only the segments that have fields related to the semantic rules are considered. For generating valid test messages, those segments that satisfy the semantic rules are regenerated and for generating invalid test messages, those segments that do not satisfy the semantic rules are regenerated (2nd-pass in Fig. 4). The remaining parts are chosen among valid segments.

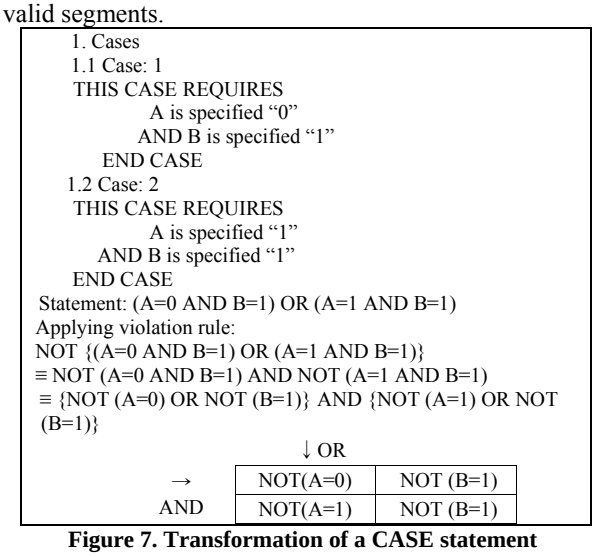

Figure 7. Transformation of a CASE statement

4. Test Requirements for VMF Tactical Data Link Messages

VMF testing confirms whether a TDS interprets VMF messages in a way that conforms to the VMF message standard. Ideally, all possible syntactic and semantic cases should be covered for VMF testing. Therefore, it would be extremely beneficial to be able to generate test messages automatically because of the evolvability, complexity, and variety of VMF messages.
In this section we derive the test requirements for VMF by analyzing the VMF message standard (MIL-STD-6017). The test requirements define what specific aspects of VMF messages should be tested. As the VMF message standard contains syntactic rules as well as semantic rules, test requirements for VMF messages are divided into two categories, test requirements for syntactic rules and those for semantic rules. Then since TDSs must be able to process not only valid messages but also invalid messages, the two categories are further divided into those for valid messages and those for invalid messages.

4.1 Test Requirements for Syntactic Rules

The smallest units composing a VMF message are the indicator fields and information fields. A field in a message is either mandatory or optional. Some information fields have range constraints according to the characteristics of their fields. A set of fields can be combined as a group. To consider all these cases, VMF testing should be conducted from the following four aspects (ASs).

- **Mandatory field existence:** detection of missing mandatory fields (AS1: all mandatory fields exist).
- **Field/Group recurrence:** confirmation whether the maximum number of repetitions of a field or a group is exceeded (AS2_1(FRI): the number of field recurrences is equal to the FRI value, AS2_2(GRI): the number of group recurrences is equal to the GRI value).
- **Field/Group presence:** validating whether field or group existence is consistent with the FPI and GPI value (AS3_1(FPI): field existence is consistent with the FPI value, AS3_2(GPI): group existence is consistent with the GPI value).

According to the above aspects for VMF testing, we derived the following test requirements (TR) for valid test messages:

- **TR1:** Messages including all possible groups and fields (AS1=true && AS2_1=true && AS2_2=true AS3_1(1)=true && AS3_2(1)=true).
- **TR2:** Messages that have a FPI value 0 and the following field of the FPI does not appear (AS1=true && AS2_1=true && AS2_2=true && AS3_1(0)=true && AS3_2=true).
- **TR3:** Messages that have a GPI value 0 and the following group of the GPI does not appear (AS1=true && AS2_1=true && AS2_2=true && AS3_1=true && AS3_2(0)=true).

We also derived the following types of test requirement groups for invalid messages based on the four aspects that we introduced at the beginning of this section:

- **TR4:** Mandatory field missing (AS1=false).
- **TR5:** The number of field recurrences is not correct (AS2_1=false).
- **TR6:** The number of group recurrences is not correct (AS2_2=false).
- **TR7:** Messages that have a FPI value 0 and the following field of the FPI exists or a FPI value 1 and the following field of the FPI does not exist (AS3_1(1)=false || AS3_1(0)=false).
- **TR8:** Messages that have a GPI value 0 and the following group of the GPI exists or a GPI value 1 and the following group of the GPI does not exist (AS3_2(0)=false || AS3_2(1) =false).

4.2 Test Requirements for Semantic Rules

A VMF message has three kinds of semantic rules for message processing: Range constraints, a CASE statement, and a CONDITION statement. For a CASE

statement, the message must satisfy at least one CASE statement and for CONDITION statements the message must satisfy all relevant CONDITION statements. Therefore, testing for the semantic rules must be conducted for the following aspects.

- **Range constraints:** validating whether the value of a field (i.e., Data Item) is within the pre-determined range (AS4: the value of a field is within the range).
- **CASE statement:** if a CASE statement exists, the message must satisfy at least one of the defined cases (AS5: a message belongs to one of the defined cases).
- **CONDITION statement:** when an 'if' clause is true, its 'then' clause must be true and if it is not, its 'then' clause can be either true or false (AS6: "if *p* then *q*").

From the above two aspects, we can derive the following groups of TRs for valid messages.

- **TR9:** Range constraint is satisfied (AS4=true).
- **TR10:** A message satisfies one of the CASE clauses (AS5=true).
- **TR11:** Both the IF-clause and the THEN-clause are true (*p*:True, *q*:True).
- **TR12:** The IF-clause is false but the THEN-clause is true (*p*:False, *q*:True).
- **TR13:** Both the IF-clause and the THEN-clause are false (*p*:False, *q*:False).

Logically, groups of TRs for the invalid messages for CASE and CONDITION statements can be derived as follows:

- **TR14:** A message does not satisfy any defined CASE statements (AS5=false).
- **TR15:** The IF-clause is true and the THEN-clause is false in the CONDITION statement (*p*:True, *q*:False).
- **TR16:** Range constraint error (AS4=false).

5. Test Generation Method

Our test generation method consists of the principles introduced in Section 3, the test requirements described in Section 4, and a test procedure. Test messages for validating the syntactic aspects are generated by combining segments after assembling slices generated by applying Principles 1 through 4. If a segment has more than one FPIs and/or FRIs, they need to be distinguished by assigning their index numbers and saved in the TDB. Slices for validating semantic aspects are generated by applying Principles 1, 2, and 4. Fig. 8 depicts our test generation procedure and shows where each of the principles is applied. In this section we illustrate our test generation method (Fig. 8) with the VMF message (K05.1) as an example.

Step 1: Dividing a message into segments

Since K05.1 (position report message) [5] is composed of four segments that belong to group indicators (G1-G4) and three segments of a sequence of fields (R1-R3) that do not belong to any group, there are seven segments altogether (applying Principle 1) (Fig. 5).

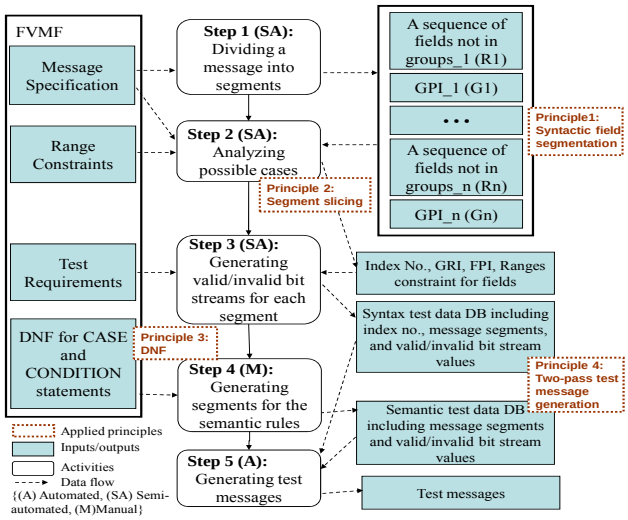

Figure 8. Test generation procedure

Step 2: Analyzing possible cases

In the case of GPI_2, there are three FPIs and no range constraints. From them we generate 15 kinds (applying Principle 2) of valid/invalid slices relevant to group GPI_2. Each slice can be substituted for its corresponding segment and after that we generate test messages by composing these segments.

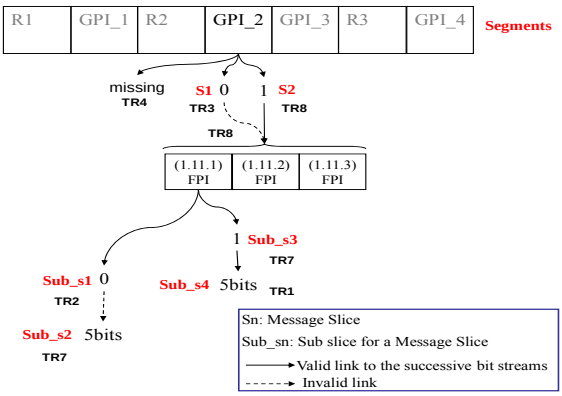

Figure 9. Message segment slicing

Fig. 9 illustrates the segment slicing procedure that is in accordance with the proposed test generation method. R1 through GPI_4 are the segments generated by Principle 1. As example message slices of the GPI_2 segment, the value of GPI_2 is 0 or 1 as shown in Fig. 9. If the value of the GPI_2 field is 0, no fields should exist within G2 groups, i.e., (1.11.1)FPI through (1.11.3)FPI. Therefore, the first slice S1 is a valid slice. However, if group G2 exists when the value of GPI is 0, the slice is a GPI error slice regardless of the validity of the remaining fields (see the link illustrated using a dotted arrow in Fig.

9). If the value of GPI is 1, the slice S2 that does not include the remaining fields of the G2 group falls under the GPI error. Fig. 9 also shows how we generate message slices satisfying the derived TRs.

Step 3: Generating valid/invalid bit streams for each segment

In Step 2, the values of presence/recurrence indicators are assigned during slicing a segment by analyzing possible cases. In this step, we consider the values of information fields included in each segment, date, time, altitude, and so on. Range constraints specified in the VMF standard are considered in this step.

Step 4: Generating segments for semantic rules

When we generate a message segment, if there is a field that has a range constraint, then message segments related to it are also generated (to satisfy TR9). For CASE and CONDITION statements, message segments are generated by referring to DNF (Principle 3) and the cause and effect information of the decision table. As we already mentioned, we apply a two-pass generation method (Principle 4) to reduce the complexity of the test message generation. Because CASE and CONDITION statements are constraints between fields in the same segment or in two different segments, we cannot generate message segments for the semantic rules without knowing the message structure. When we generate message segments for semantic rules using decision tables derived from DNF information of FVMF, if a cause and effect field is included in different segments, we add an additional annotation to each slice to be used when we compose them later.

Fig. 10 shows an example of a decision table for generating segments for the CONDITION statements. The 'ORIGINATOR ENVIRONMENT/CATEGORY' field (275/001) among the CONDITION statements of K05.1 has four cases. Each row of Fig. 10 is related to valid message segments. Invalid message segments can also be generated based on this table. Four valid segments and at least four invalid segments can be generated for the CONDITION statement related to 275/001 field.

	1	2	3	4	
(1.13)275/001	0	1	2	3	R3 segment
(1.14.1)FPI	0	0	0	V	GPI_4 segment
(1.14.2)FPI	V	0	0	0	
(1.14.3)FPI	0	V	0	0	
(1.14.4)FPI	0	0	V	0	

V: do not matter

TR11 test case

Figure 10. Decision table for 'ORIGINATOR ENVIRONMENT/CATEGORY' field

However, as for K05.1, there is no condition that does not satisfy the IF-clause because two bits are assigned to the 'ORIGINATOR ENVIRONMENT/ CATEGORY' field and conditions for all possible values (0 through 3) are defined. Therefore, we do not need to consider TR12 and TR13.

Step 5: Generating test messages

Test messages are generated by combining the segments produced through Steps 1 – 4. Each segment is stored with its type information, i.e., 'valid' or 'invalid'. For an invalid type segment, its error type should be stored as well.

6. Test Generation Tool and Evaluation

We implemented the test generation method in a test generation tool, which is a part of the comprehensive VMF test tool that we are currently developing. The test generation tool first reads the syntactic and semantic rules of the VMF message specified in the FVMF. This information is saved in the test database (TDB) so that the procedure need not be repeated for the same messages later. Then the test generation tool generates message segments, including slices. This process is depicted in Fig. 11.

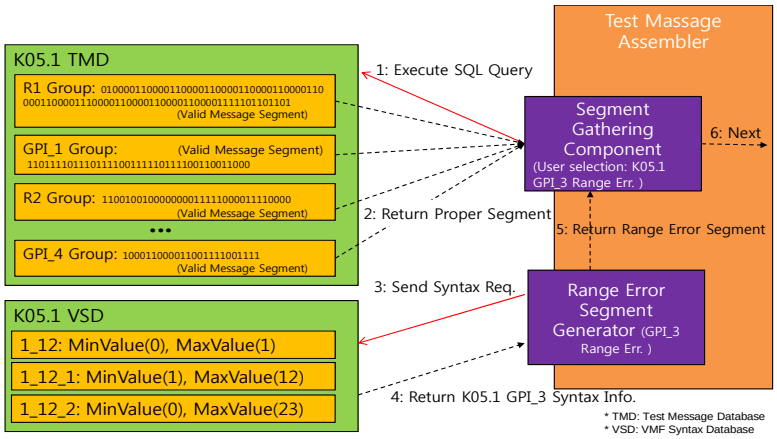

Figure 11. Message segment generation

In order to provide adequate information about the message segment, each message segment in the TDB contains, in addition to the bit stream (Value), the name of a group segment (NumGroup), valid/invalid information, the error type (Code), and descriptions (Comments). It also includes the index number field necessary for distinguishing the FPIs within a segment. Fig. 11 shows bit stream examples for message segments and how they are composed to make a test message within the VMF test tool. A Segment Gathering component gathers segments and combines them automatically. Once the user selects the types of test messages, the Segment Gathering component sends an SQL query and extracts the right segments from the TDB. Segments related to range constraints are generated through the Range Error Segment Generator component.

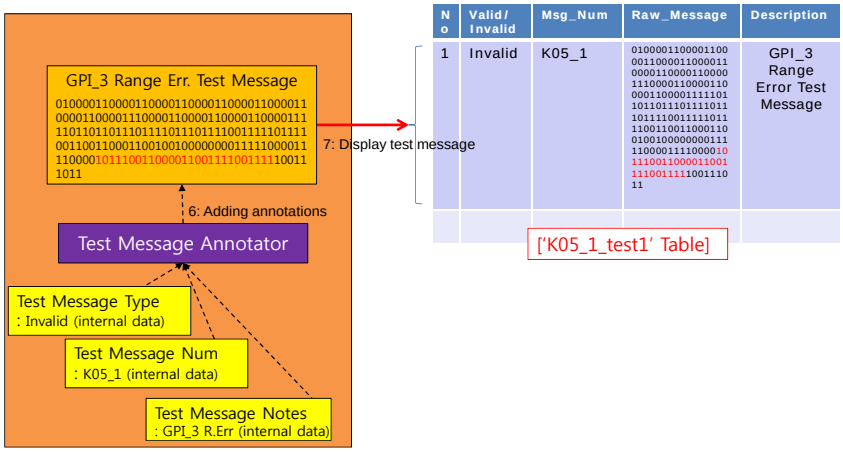

Figure 12. Test message example

Fig. 12 illustrates an example test message generated in accordance with the user's specification of an error type. The Test Message Annotator adds annotations to a message, and then the Test Generator sends the message to the Message Monitoring Tool, a part of our VMF test tool, after assembling additional information related to the generated test message. The example test message (K05.1) is a case where the error type is a range error of a field within the GPI_3 segment.

The test generation tool generates valid test messages by composing a valid message segment or generates invalid test messages by composing a specific error segment with valid message segments. Test messages generated in this way are stored in the TDB of our VMF test tool.

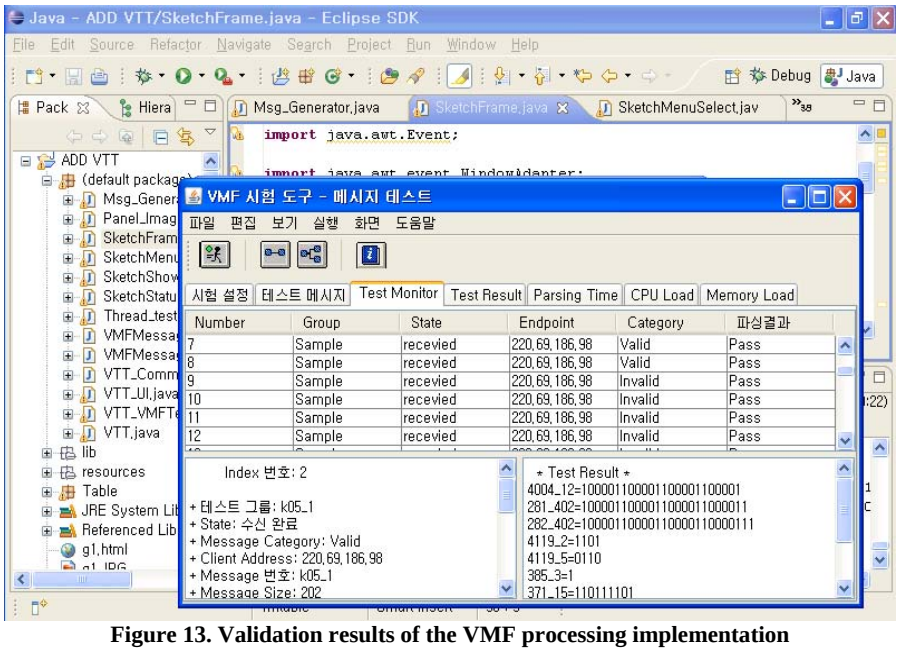

Figure 13. Validation results of the VMF processing implementation

Fig. 13 shows an operation of the VMF test tool for testing a VMF processing implementation. The method presented in this paper was implemented as its Test Generator component. Test messages are interpreted by the VMF processing implementation and then the decision whether or not it is valid is displayed together with its error type if there is an error. The test messages generated in our method were used to validate the various versions of a VMF message parser. They were validated in advance by using test messages generated by the hand-coding method, which was used by the VMF parser generator development team for validating auto-generated parsers for each VMF message. For generating specific types of test messages, the required code has to be manually added in the hand-coding method. We shared the derived test requirements beforehand so that the parser development team could refer to them. We found, however, that the hand-coding method missed some syntactic rules (e.g., the maximum number of group recurrence and segmentation faults) whereas our method covered all types of test messages covered by the hand-coding method. This implies that our method provides an efficient and convenient way to generate test messages that are defined in the test requirements. In our tool, the generated test messages or segments are saved in the TDB so as not to regenerate the same test messages. In particular, with our method it is possible to control the number of message validations for each execution and to adjust test coverage by varying the way of combining message segments.

So far we have introduced our test generation approach for VMF Tactical Data Link Messages. With the proposed approach, the two challenges of VMF Tactical Data Link Messages – coping with variable message length and handling semantic message rules – were successfully overcome. The segmentation approach made it simple to apply the existing software testing techniques. Furthermore, the evaluation results revealed that segmentation and message slicing can be conducted semi-automatically or automatically. The test generation tool requires user intervention only to enter the type of messages – valid/invalid message - and the error type when a user chooses an invalid message. A user can also generate test messages randomly and in this case the test generation tool provides the correct answers of interpreting the test messages. Most of all, during the iterations for generating and validating test messages in accordance with the test generation approach and test requirements, there were cases of missing test requirements. With our approach existing slices could be used in many cases to generate the relevant test messages for the missed test requirements. Together with our client (the Korean Agency for Defense Development), we validated our tool using three representative parsers developed by the parser generation team and the tool was successfully used to detect errors and improve the generated parsers.

Through the evaluation, we found that our method is effective in reducing complexity due to message length variability and semantic rules, and it will be helpful but difficult to automate the test message generation for the following aspects:

- **Range constraints:** Because information fields related to range constraints and their ranges exist in the TDB whenever an information field appears, we have to refer to the TDB to check whether it has range constraints. It is possible to do this with hard-coding, but then we would have to change the code to reflect changes of a VMF message including range constraints. Therefore, we manually generate slices related to range constraints.
- **Group recurrence:** Since group recurrence can be considered after the bit stream value of a segment has been generated and a message can contain nested groups, the complexity of segmentation may be too high to control. Therefore, we manually generate slices for the segments that have group recurrence.
- **CASE and CONDITION statements:** Because CASE and CONDITION statements specify the relationships between fields that belong to different segments, it is difficult to automatically generate slices constrained by them.

These aspects make it difficult to generate test messages in real time, so we generate all possible segments for all messages in advance to test VMF parsers in near real time. This has an advantage in reusing test messages but when new VMF messages are added or the existing messages are changed it is difficult to cope with such situations rapidly. When a message is changed, some segments should be generated to reflect that change. Since the military context needs rapid validation of TDSs related to a newly added message, it is not easy to address that need with our approach. It is conceivable, though, to enforce a function such that a user can enter the real value of an indicator or a field. It is possible to extend the test generation tool so that it is possible to cope with newly added or changed messages easily. However, this problem is an inherent limitation in any semi-automatic method and can be improved with more automation support.

7. Conclusions

This paper presented our method and evaluation for test generation of variable length messages for the tactical data link systems. The purpose of test generation is to validate the ability of TDSs to correctly interpret VMF messages as specified in the VMF standard and report the error types that a VMF message contains in the case of receiving an invalid message. Test generation for VMF messages is nontrivial because message length is variable, as the fields such as GPI, GRI, FPI, and FRI,

can be combined in a multitude of ways to produce messages with various lengths. Moreover, a VMF message includes semantic processing rules that do not allow straightforward test generation.

The presented test generation method consists of the principles that we used for meeting the complexity of testing VMF messages, test requirements derived by analyzing the characteristics of VMF messages as specified in the standard, and a procedure for test generation based on them. Test requirements are the basis of test message generation. By applying the proposed test generation principles we generated test messages that cover the defined test requirements. Through the message segmentation and segment slicing principles we addressed the message length variability challenge. By using DNF for uniform representation and using two-pass test message generation, we addressed the challenges of the semantic rules. Syntactic field segmentation, segment slicing, and two-pass test generation principles were effective in reducing the complexity of variation by promoting automation of test messages generation. DNF was effective in clarifying and simplifying the semantic rules such as constraints between fields and helped reduce the potential misinterpretation.

From the perspective of efficiency, the test coverage is an essential issue in our method and we were able to control the coverage level through the segmentation principle. Each segment has a number of valid/invalid slices that satisfy the test requirements. Basically, for each test requirement at least one test case is generated and therefore each valid and invalid slice is included in some test case at least once. By including all possible combinations of slices we can reach the maximal coverage level. Test messages are generated in an on-demand manner by combining message slices.

We implemented the presented test generation method as a part of our VMF test tool and used it for validating the various versions of a VMF message parser. By using the test messages generated from the presented test generation method we were able to find subtle faults such as violation of the maximum number of group recurrence, which was missed when the VMF parser was validated with the test messages produced from a hand-coding method. In addition, it is efficient in that it makes it possible to automate some of the steps for generating test messages.

In the current version of our test generation tool, some steps were conducted manually, so further research is needed for dealing with test message generation for the newly added or changed VMF messages in near real time. We plan to implement the idea of including FPI and FRI also as slices of segments, thereby more fully realizing the automation of the VMF test generation process. In addition, our research for classifying message length variability patterns in communication protocols and developing protocol independent test requirements patterns are in progress.

References

[1] Department of Defense, MIL-STD-6017, Variable Message Format, 2004.

[2] Department of Defense, “Connectionless Data Transfer Application Layer Standard”, Military Standard MIL-STD- 2045-47001D, US, 2005.

[3] REDONDO SYSTEMS INCORPORATED, http://www.redondosystems.com/ vmf.htm, [30 September 2009].

[4] J.G. Kim, J.H. Lee, and S.W. Kang, “Formalized Variable Format Message for Efficient Parsing”, Proc. of 2006 KICS Fall Conference, Vol. 34, 2006.

[5] C.O. Lee, J.S. Song, B.W. Kang, D.S. Han, and W.K. Lim, “An Automatic Generating Framework for Bit Message Parser”, Proc. of 2006 KICS Fall Conference Vol. 34, 2006.

[6] B. Beizer, Software Testing Techniques, 2nd ed., Van Nostrand- Reinhold, 1990; Ch12.

[7] Variable Message Format - VMF, http//www.lm-isgs.co.uk/def, [30 September 2009].

[8] Variable Message Format (VMF) Testing Program, http://jitc.fhu.disa.mil/brochure/ vmf.pdf, [30 September 2009].

[9] K. H. Rosen, Discrete Mathematics and its Applications, 5th ed., McGraw Hill, 2003; Ch1.

[10] E. Mak and S. Mittal, M.H. Hwang, “Automated Link 16 Testing using DEVS and XML”, Journal of Defense Modeling and Simulation JDMS, accepted 2007, http://www.acims.arizona.edu/PUBLICATIONS/PDF/AutomatedTestingPaper.pdf, [30 September 2009].

[11] R. Lai, “A survey of communication protocol testing,” Journal of Systems and Software, Vol. 62, Issue 1, 1 May 2002; 21-46.

[12] Kang, S., Seo, Y., Kang, D., Hong, M., Yang, J., Koh, I., Shin, J., Yoo, S., and Kim, .M. “Development and Application of ATM Protocol Conformance Test System”, Proc. of the IFIP Tc6 12th international Workshop on Testing Communicating Systems: Method and Applications, 1999 ; 331-346.

[13] D. Lee and M. Yannakakis, “Principles and methods of testing finite state machines - A survey”, Proc. of IEEE, 1996.

[14] S. Milliner and A. Delis, “Networking abstractions and protocols under variable length messages”, Proc. of the 3rd International Conference on Network Protocols (ICNP'95), 1995.

[15] Y. T. Wu, J. F. Chang, "Collision resolution for variable length messages", IEEE Transactions on Communications, Vol. 41, Issue 9, 1993; 1281 - 1283.

[16] J.G. Kim, S.W. Kang, N.B. Lee, “Reliability Assessment of Translated VMF/Link-16 Messages,” Proc. of the 19th IEEE International Symposium on Software Reliability Engineering (ISSRE '08), Nov 11-14, 2008.

[17] G.J. Myers, Revised and updated by T. Badgett and T. Thomas, with C. Sandler, The Art Of Software Testing, 2nd ed., John Wiley & Sons, Inc., 2004; Ch4.