An Example of a Language Representation based on C-BML

Ľubomír Dedera[1], Marek Benčík[2]

[1]Armed Forces Academy of Gen. M. R. Štefánik, Department of Informatics, Demänová 393, 031 06 Liptovský Mikuláš, Slovak Republic
[2]Base of Mobile Communications and Information Systems, Zarevúca 2, 034 01 Ružomberok, Slovak Republic
Lubomir.Dedera@aos.sk, Marek.Bencik@mil.sk

Abstract. In this paper we present a case study on creating language representations and corresponding language processors in the area of military C2 systems utilizing the standardized C-BML data structures. We point out how techniques and tools used in the area of compilers are exploitable in integration of national command and control systems and deployment in multinational environment.

Keywords: C-BML, Domain-Specific Language, Grammar, Syntax, Semantics, Language processor.

1 Introduction

Computer languages (CLs) based on strict formal syntax and semantics could be utilized in current military C2 systems in various ways, e.g.:

- as a means for information interchange and integration between various kinds of military systems;
- as specification and programming languages dedicated for military domains.

Their potential in this field comes from the fact that military application domains have established their own terminology with quite formal syntax and semantics [1] as well as standardized way of information exchange [2]. Probably the most significant initiative in this field within NATO is a series of activities and projects connected with the development of the Coalition Battle Management Language (C-BML) [3], [4], [5], [6], which resulted in establishing of the standard [7]. The objective was to "define an unambiguous language to describe a commander's intent, to be understood by both live forces and automated systems, for simulated and real world operations. The resulting language is intended to be applicable not only to simulation systems, but also to operational command and control systems, and robotic systems" [3].

C-BML can be considered as an example of domain-specific language (DSL) designed specifically for the military domain. As the basic principle for the C-BML the paradigm of "5W" (Who-What-Where-When-Why) has been chosen and a series of experiments and demonstrations has been published so far.

The process of development of the C-BML standard has been divided into three phases [7]:

- Phase 1, Data Model: Describes "a sufficient data model to unambiguously define a set of military orders using JC3IEDM as a starting point and extending it as necessary so that the orders can be interpreted by C2, M&S, and ultimately autonomous systems." This standard describes the data model as a subset of JC3IEDM [2] and specifies the information exchange content and structure in the form of an Extensible Markup Language (XML) schema.
- Phase 2, Formal Structure (Grammar): The result should introduce "a grammar (syntax, semantics, and vocabulary) as part of the information exchange content and structure specification. The objective is to formalize the definition of tasks such that they are rigorous, well documented, and parseable."
- Phase 3, Formal Semantics (Ontology): Phase 3 will "include development of a battle management ontology to enable conceptual interoperability across systems."

Phase 1 has been finished with creation of

- standard data model and procedures for extending this data model, and
- description of the information components of the language.

This specification does not provide the structure and content for all C-BML expressions that may be needed by the broad user community, but provides an initial set of constructs that can be used to create initial implementations of the standard. Phase 2 should provide a formal grammar for plans, orders, requests, and reports and Phase 3 formal semantics of the language [7].

From the development of the C-BML grammar point of view experiments and demonstrations with Command and Control Lexical Grammar (C2LG) with a GUI editor [4][5] has been undergone with the aim to prove that C-BML is a suitable tool for the exchange of orders and reports between C2 systems and constructive simulators [6]. The paper [8] aims at the way how to implement lexical functional grammar based approaches into object-oriented class hierarchies with the conclusion that "Lexical Functional Grammar approach combined with object-oriented representation is a good practice in order to represent grammar in BML."

The utilization of CLs within military command and control (C2) systems has been also discussed in [8], [9]. In [8] the notion of DSLs in the context of C2 systems has been introduced and the parallels between programming languages and DSLs for C2 systems regarding their syntactical and semantic processing have been pointed out. A simple example of using DSLs in a C2 system has been given, where the role of a trained military commander in relation to a DSL can be analogical to the role of a computer programmer in relation to a programming language. In [9] the role of computer language syntax and semantic processing techniques as a means to achieve semantic interoperability has been stressed out and the notion of DSLs with multilingual support in the context of C2 systems has been introduced. As a main means to achieve semantic interoperability and multilingual support, the techniques of abstract and concrete syntax and semantic processing have been proposed. The ideas behind abstract syntax and semantic processing are similar to the ones presented in [8], where the abstraction and association relationships between classes are used to represent model of the grammar. Due to national information systems integration and

interoperability challenges we find it interesting to study and design DSLs with different concrete syntaxes, but with mutually related semantic processing.
This contribution could be considered as a contribution to a discussion regarding the development of a formal grammar of C-BML (Phase 2 of the project). In the next sections and paragraphs we give an example of a parseable context-free grammar describing simplified structure of a language for the control of combat operations intended for Slovak (human) environment (so-called Slovak language representation, SLR). Next we are going to pay attention to the process of creation of a language processor for the above mentioned language representation that would be able to transform sentences of SLR into Phase 1 – standardized XML data structures of C-BML. In the conclusion we sum up our recommendations regarding the principles of abstract and concrete syntaxes and handling of semantic processing on both abstract and concrete levels.

2 C-BML Data Model

As a central reference model for C-BML data model, JC3IEDM [2] has been chosen. It is sufficiently robust to cope with the amount of data that should be interchanged among systems for which C-BML is proposed (C2, robotic, M&S). The model is based on a set of concepts, their attributes, relations and business rules to check data consistency and is described using XML schemas.

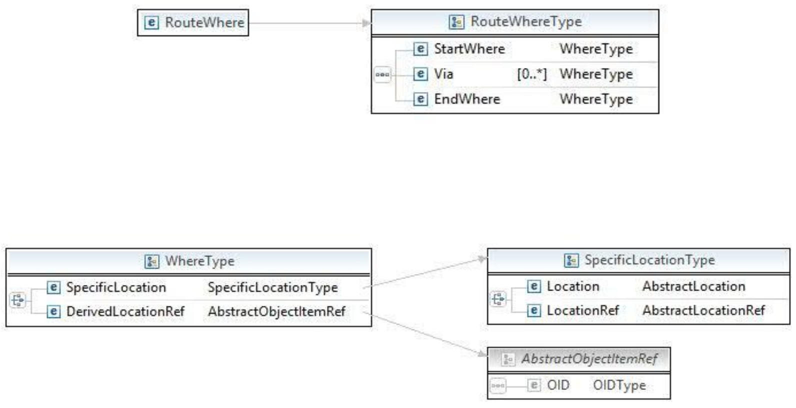

Fig.1. C-BML data structures

The main information components of C-BML could be described by the 5W (Who-What-Where-When-Why) rule. Fig. 1 illustrates the component Where describing a route to be followed in an action (RouteWhere).

3 Grammar for Slovak Language Representation

In general, syntax of computer languages can be described by means of context-free grammars (CFGs) or from CFGs derived Backus-Naur forms (BNFs). [11]. A context-free grammar is a 4-tuple G = (N, T, P, S), where N is a finite set of nonterminals (or variables, depicted using <…>), T is a finite set of terminals (lexical elements, depicted in **bold**), S, $S \in N$ is the starting symbol of the grammar from which each derivation starts, and P is a finite set of productions (or rewriting rules) of the form $B \rightarrow \alpha$, where $B \in N$ is the left-hand side and $\alpha \in (N \cup T)^*$ is the right-hand side of the production.

Our experimental grammar for SLR consists of 25 rewriting rules. The names of relevant nonterminals reflect the 5W principle. Rules 1-2 describe the sequence of commands. Rule 3 describes a structure of a single command. Rules 4-6 describe 3 particular tasks – attack (útoč), march (pochoduj), and occupy (obsaď). Rules 7-9 describe taskers and taskees, rules 10-19 time data, rules 20-21 not compulsory Why data and rules 22-25 Where data. Starting symbol is <commands>.

1. <commands> → <command> <commands>
2. <commands> → Ɛ
3. <command> → <TaskerWho> **prikazuje** <TaskeeWho> <TaskWhat> <StartWhen> <EndWhen> <Why>**;**
4. < TaskWhat > → **útoč** <AtWhere>
5. < TaskWhat > → **pochoduj** <RouteWhere>
6. < TaskWhat > → **obsaď’** <AtWhere>
7. < TaskerWho > → <unit>
8. < TaskeeWho > → <unit>
9. <unit> → **string**
10. < StartWhen > → <time data>
11. < time data > → **o** <time>
12. < time data > → **najneskôr o** <time>
13. < time data > → **ihneď’, nie neskôr ako** <time>
14. < time data > → **po** <time>
15. < time data > → **ihneď’ po** <time>
16. < time data > → **pred** <time>
17. < time data > → **nie pred** <time>
18. <time> → **time**
19. < EndWhen > → <time data>
20. <Why> → **string**
21. <Why> → Ɛ
22. <AtWhere> → **coordinates**
23. <RouteWhere> → **z** <AtWhere> **do** <AtWhere> <via>
24. <via> → **cez** <AtWhere> <via>
25. <via> → Ɛ

Regarding lexical elements, some of them represent a unique sequence of characters (**útoč**, **inheď’**, **o**, **z**, **cez**, etc.), others (namely **string**, **time**, **coordinates**) cover sets of data of a particular format.

4 Language Processor

In the next step a language processor for the grammar for SLR has been developed. Its role is to transform syntactically valid expressions in SLR into standardized C-BML data structures. For example, the command

"3.Army" prikazuje "1.Division" útoč 12.456987 21.548796
o 29.01.2013 15:00:02 po 29.01.2013 15:00:03;

is transformed by the language processor into:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <CBML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.sisostds.org/schemas/c-bml/1.0"
xsi:schemaLocation="http://www.sisostds.org/schemas/c-bml/1.0 ../../example-
expressions-schema/example-cbml-expressions.xsd">
    <Task>
    <What>
        <ActionTask xsi:type="OtherActionTask">
            <ActivityCode>ATTACK</ActivityCode>
        </ActionTask>
    </What>
    <TaskerWho>
        <OrganisationRef xsi:type="UnitRef">
            <OID> "3.Army"</OID>
        </OrganisationRef>
    </TaskerWho>
    <TaskeeWho>
        <OrganisationRef xsi:type="UnitRef">
            <OID> "1.Division"</OID>
        </OrganisationRef>
    </TaskeeWho>
    <AtWhere>
        <SpecificLocation>
            <Location xsi:type="GeographicPoint">
                <LatitudeCoordinate>12.456987</LatitudeCoordinate>
                <LongtitudeCoordinate>21.548796 </LongtitudeCoordinate>
            </Location>
        </SpecificLocation>
    </AtWhere>
    <When>
        <StartWhen>
            <AbsoluteTime>
                <SpecifiedTime>
                    <Datetime>20130129150002</Datetime>
                    <StartQualifierCode> AT </StartQualifierCode>
                </SpecifiedTime>
```

```
          </AbsoluteTime>
     </StartWhen>
     <EndWhen>
          <AbsoluteTime>
               <SpecifiedTime>
                    <Datetime>20130129150003</Datetime>
                    <EndQualifierCode> AFT </EndQualifierCode>
               </SpecifiedTime>
          </AbsoluteTime>
     </EndWhen>
</When>
</Task>
 </CBML>
```

The language processor has been constructed using software tools Flex/Bison that were developed primarily for compiler constructions. The tool Flex was used for the construction of the lexical analyzer. This tool requires lexical elements to be specified with regular expressions. The parser has been created using the Bison tool. This tool requires the description of the context-free grammar for the source language (SLR) combined with the semantic action routines (in the C programming language). The tool itself offers support for the information exchange among semantic routines by means of semantic records and a semantic stack. In our case the main role of semantic routines was to collect information from lexical elements and/or nested structures and output this information in the form of XML data structures as it has been shown in the above example.

4 Conclusion

In this paper we have given an example of a concrete language representation for the C-BML language together with the description of its language processor that has been developed using the tools Flex/Bison.

By this example we wanted to point out how it is possible to construct concrete language representations tailored for particular environments. Different environments can represent either different nations and/or the fact that the language should be used primarily for communication with humans (either in graphical or in textual form) or machines. For this reason we find it interesting to study and design languages with different concrete syntaxes, but with mutually related semantic processing. The principles of language design and processing presented in the paper could lead to the design of the whole family of "BMLs", each tailored for particular audience or purpose, together with their language processors. Considering the principles C-BML has been designed on it seems to be possible. This goal can be accomplished by utilizing an abstract syntax of the language and defining the great majority of semantic processing on it.

In our work we also have come to 2 conclusions:

- We have shown that computer languages and techniques of their processing can be utilized in the area of semantic interoperability (e.g., data and command convrsions) among different military systems.
- We have shown that the tools Flex and Bison that have been originally developed for the support of compiler construction can be utilized in a completely different application domain.

The grammar described in the paper was designed for demonstration purposes only and should not be considered as a result of deep research in this area.

References

1. NATO STANAG 2014: Formats for orders and designation of timings, locations and boundaries, North Atlantic Treaty Organisation, 2000.
2. NATO STANAG 5525: Joint C3 information exchange data model – JC3IEDM, North Atlantic Treaty Organisation, 2007.
3. C. Blais, M. R. Hieb, and K. Galvin, K., "Coalition battle management language (C-BML) study group report," 05F-SIW-041, Fall Simulation Interoperability Workshop 2005, Orlando, FL, September 2005.
4. U. Schade and M. R. Hieb, "Formalizing battle management language: A Grammar for Specifying Orders," Spring Simulation Interoperability Workshop, Huntsville, Alabama, April 2006.
5. K. Rein, U. Schade, and M. R. Hieb, "Battle management language (BML) as an enabler," IEEE International Conference on Communications, ICC 2009, Dresden, Germany, June 2009.
6. K. Heffner, A. Brook, N. de Reus, L. Khimeche, O. M. Mevassvik, M. Pullen, U. Schade, J. Simonsen, and R. Gomez-Veiga, NATO MSG-048 C-BML final report summary, 2010 Fall Simulation Interoperability Workshop (Paper 10F-SIW-039), September 2010, Orlando, FL.
7. SISO-STD-011-2014: Standard for Coalition Battle Management Language (C-BML) Phase 1. Simulation Interoperability Standards Organization, Inc., 2014.
8. Gustavsson, P. M., Wemmergard, J., Jonsson, F., Object-Orientated Implementation of Grammar Based Battle Management Languages, Spring Simulation Interoperability Workshop (Paper 12S-SIW-001), March 2012, Orlando, FL.
9. Dedera, Ľ.: Domain-Specific Languages for Command and Control Systems. In: Science&Military, No 1, Vol. 5, pp. 40—46 (2010)
10. Dedera, Ľ.: Semantic Interoperability by Means of Computer Languages. In: Military Communications and Information Technology: A Trusted Cooperation Enabler, Vol. 1, Military University of Technology, Warsaw, pp. 209-220 (2012)
11. M. Fowler, Domain-specific languages, Addison-Wesley Professional, Boston, 2010.