Applying A Formal Language of Command and Control for Interoperability Between Systems

Presented to the AFCEA - GMU C4I Center Symposium on “Critical Issues in C4I”

Dr. Michael Hieb
George Mason University
US
mhieb@c4i.gmu.edu

Dr. Ulrich Schade
FGAN-FKIE
GERMANY
schade@fgan.de

Content

1. The Development of a Formal Grammar
2. Designing a Command and Control Grammar
3. A Tasking Grammar
4. Related Grammar Developments
5. Implementations
6. Outlook
A Linguistic Basis for A Computational C2 Grammar

We have developed a formal language for military communication (including formal communication of intent) because not all recipients can understand free text expressions. Examples are:

- Coalition Forces not speaking English as their native tongue
- Simulated Forces
- Future (smart) Robotic Forces

Formal Language

- Formal Languages provide a rigorous framework for automated processing.
- Formal languages are defined by grammars.
- The military domain provides excellent structure to terms and actions in a formal language.
- Current Message and Data-based communications do not go far enough – a grammar is needed to give additional meaning.
Orders and reports are not “formally” represented in the current data models like the Joint Coordination, Command and Control Information Exchange Data Model (JC3I EDM).

• In order to communicate one needs a language.
• Current Data Models are not a language; especially, they do not give meaning to the tasks.
• A language needs a lexicon (this can be provided by data models).
• It also needs a grammar (to concatenate the lexical items) and give meaning to the catenation.

A formal language is defined by a grammar. The grammar provides

• a lexicon in order to determine the words which may be used as well as their semantics (their meaning);
• a finite set of rules in order to determine how to concatenate the words and to give meaning to the catenations.
Lexical Functional Grammar

Lexical Functional Grammar (LFG) is a theory of grammar – that is, in general terms, a theory of:

• syntax (how words can be combined together to make larger phrases, such as sentences)
• morphology (how morphemes - parts of words - can be combined to make up words),
• semantics (how and why various words and combinations of words mean what they mean), and
• pragmatics (how expressions are used to transmit information)

We use the Lexical Functional Grammar as the basis for the Formal Grammar.

An Extensive Literature on LFG

http://www.essex.ac.uk/linguistics/LFG/

A Sample

Bresnan, Joan. 1972.
Theory of complementation in English syntax.
Ph.D. thesis, MIT.

Bresnan, Joan (editor). 1982b.
The Mental Representation of Grammatical Relations.
Cambridge, MA: The MIT Press.

Things are not always equal.
Lecture Notes in Computer Science, Volume 2588.

Dalrymple, Mary. 2001.
Lexical Functional Grammar, volume 34 of Syntax and Semantics.

1148 Entries in LFG Bibliography!
We developed our C2 Grammar such that it includes Command Intent, Tasking and Coordination.

**Tasking** → Command(Intent) OB* Coord_Space* Coord_Time*

**Command Intent** → [Expanded Purpose] [Key Tasks] [End State]

OB is a basic order expression by which tasks are assigned to units. OB consists of a tasking verb and constituents.
A BML Tasking Grammar

The production rules for the basic expressions have the following general form:

\[ B \rightarrow \text{Verb Tasker Taskee (Affected | Action)} \]
\[ \phantom{B} \quad \text{Where Start-When (End-When) Why Label (Mod)*} \]

“Verb” is an action, normally a task;
“Tasker” is a “Who”, the unit which commands the task;
“Taskee” is a “Who”, the unit which executes the task;
“Affected” is a “Who”, the unit which is affected by the task;
“Action” is another action/task affected by the task;

“Where” is a “location phrase”;
the “When”s are “time phrases”;
“Label” is a label given to the task in order allow it to be referred in other basic expressions.
The production rules for basic expressions have the following general form:

\[ B \rightarrow \text{Verb Tasker Taskee (Affected | Action)} \]

\[ \text{Where Start-When (End-When) Why Label (Mod)*} \]

Whether there is “Affected” or “Action” is determined by the verb. This is indicated by the round brackets. The Verb also determines the kind of Where (At-Where or Route-Where) to be used.

Why represents a reason for the task – the mission’s purpose. FM 3-90 [USA, 2001] offers a list of verbs to express the Why, namely divert, enable, deceive, deny, prevent, open, envelope, surprise, cause, protect, allow, create, influence, and support. We will label these verbs “purpose-verbs”. From a linguistic perspective, the verbs can be divided into three groups, namely

1) those that can be used with an argument that is an object, like “in order to deceive the enemy”,
2) those that cause a state, and
3) those that need another task as argument, like “in order to enable task DELTA”.

Why → in-order-to PVerb (Who | Task)

Why → in-order-to cause (EndState)

Why → in-order-to enable (Task)
### A BML Tasking Grammar

#### Rules for basic expressions (examples)

("verbs" are taken from JC3IEDM-table "action-task-category-code")

<table>
<thead>
<tr>
<th>Rule</th>
<th>Verb</th>
<th>Tasker</th>
<th>Taskee</th>
<th>Route-Where</th>
<th>Start-When</th>
<th>(End-When)</th>
<th>Why</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>B →</td>
<td>advance</td>
<td>Tasker</td>
<td>Taskee</td>
<td>Route-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
<td>Label</td>
</tr>
<tr>
<td>B →</td>
<td>ambush</td>
<td>Tasker</td>
<td>Taskee</td>
<td>Affected</td>
<td>At-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
</tr>
<tr>
<td>B →</td>
<td>assist</td>
<td>Tasker</td>
<td>Taskee</td>
<td>Action</td>
<td>At-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
</tr>
<tr>
<td>B →</td>
<td>attack</td>
<td>Tasker</td>
<td>Taskee</td>
<td>Affected</td>
<td>Route-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
</tr>
<tr>
<td>B →</td>
<td>block</td>
<td>Tasker</td>
<td>Taskee</td>
<td>Affected</td>
<td>At-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
</tr>
<tr>
<td>B →</td>
<td>defend</td>
<td>Tasker</td>
<td>Taskee</td>
<td>(Affect.)</td>
<td>Route-Where</td>
<td>Start-When</td>
<td>(End-When)</td>
<td>Why</td>
</tr>
</tbody>
</table>

#### Rules for constituents (examples)

<table>
<thead>
<tr>
<th>Rule</th>
<th>At-When</th>
<th>Route-Where</th>
<th>Source</th>
<th>Destination</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start-When</td>
<td>→</td>
<td>start</td>
<td>Qualifier1</td>
<td>Point_in_Time</td>
<td></td>
</tr>
<tr>
<td>Start-When</td>
<td>→</td>
<td>start</td>
<td>Qualifier2</td>
<td>Action</td>
<td></td>
</tr>
<tr>
<td>Qualifier1</td>
<td>→</td>
<td>AFT, ASAP, ASAPNI, AT, BEF, NLT, NOB</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

JC3IEDM-table "action-task-start-qualifier-code"

### A BML Tasking Grammar

#### Rules for constituents (examples, continued)

<table>
<thead>
<tr>
<th>Rule</th>
<th>At-Where</th>
<th>Route-Where</th>
<th>Source</th>
<th>Destination</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>at</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( Source )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( Path )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>along</td>
<td>Route</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>towards</td>
<td>Direction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>from</td>
<td>Location</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>to</td>
<td>Location</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>via</td>
<td>Location*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FGAN TIE
In the same way, we develop a formal reporting grammar. We differentiate
- reports about military tasks
- reports about events
- reports about status
- reports about positions

Rule forms for basic report expressions (RB):

\[
\begin{align*}
\text{RB} & \rightarrow \text{Task-Report} \text{ Verb Executor (Affected|Action)} \\
& \quad \text{Where When (Why) Certainty Label (Mod)*}
\end{align*}
\]

\[
\begin{align*}
\text{RB} & \rightarrow \text{Event-Report} \text{ EVerb (Affected|Action)} \\
& \quad \text{Where When Certainty Label (Mod)*}
\end{align*}
\]

\[
\begin{align*}
\text{RB} & \rightarrow \text{Status-Report} \text{ Hostility Regarding (Identification Status-Value)} \\
& \quad \text{Where When Certainty Label (Mod)*}
\end{align*}
\]

(Position Reports are expressed in the form of Status Reports.)
**Command Intent**

Cl $\rightarrow$ [Expanded Purpose] [Key Tasks] [End State]

The Expanded Purpose is similar to the End State, but expresses more general aspects of the resulting situation.

The Key Tasks are tasks and conditions that are essential to accomplishing the mission.

The (desired) End State describes the resulting situation that is achieved when the mission is accomplished.

---

**C2LG Implementation**

C2LG is being used in an effort called the “Battle Management Language” (BML)

BML is being developed as:

- A Standardized XML Schema supported by
  - a set of Web Services
  - standard semantics
- A Formal Grammar (C2LG)
An Implementation of the Tasking Grammar

Development of a Company Patrol Order

Patrol Order C2LG Expression

OB → patrol Tasker Taskee Route-Where
Start-When (End-When) Why Label (Mod)*

patrol 3Kp_PzGrenBtl332 1Zug_3Kp_PzGrenBtl332
along [base1_PzGrenBtl332, patrolRouteCheck4,
patrolrouteCheck8, controlPoint1, controlPoint3, controlPoint6,
patrolRouteCheck3]
start AFT 291341ZJAN07 end AT 291541ZJAN07
deny
patrol-1170074465084
System Architecture of the Demonstration presented by NATO MSG-048 at I/ITSEC, Orlando, Nov. 2007

C2LG Papers – Widely Recognized

April 2006 - On the Conference “Recommended Reading List”

June 2006 - Nominated for Best Paper

April 2007 - On the Conference “Recommended Reading List”

June 2007
Conclusions

- We have presented a formal language for conducting operations through space and time.
- The language described is designed explicitly for supporting automated Command and Control Applications.
- The language presented includes mechanisms to support representing Command Intent.
- The grammar this language is based on is being developed and standardized in NATO and IEEE.
- The use of the language not only enables decision support, but also supports collaboration and agility.

Thanks for Your Attention!

Questions and Comments are appreciated.
**Elaboration of Themes**

Patient < Essence; Ptnt(Process,Physical).
An essential participant that undergoes some structural change as a result of the event.

Theme < Essence; Thme(Situation,Entity).
An essential participant that may be moved, said, or experienced, but is not structurally changed.

---

**BML Reporting Grammar**

Task Report Expressions are similar to Order Expressions, besides

- they do not include a Tasker;
- instead of Taskee, there is an Executer;
- they – like all Report Expressions – include Certainty.

- Certainty → RPTFCT (= reported as fact)
- Certainty → RPTPLA (= reported as plausible)
- Certainty → RPTUNC (= reported as uncertain)
- Certainty → IND (= indeterminate)

(Certainty values are taken from JC3IEDM’s table “reporting-data-credibility-code.”)