# Command and Control Lexical Grammar (C2LG) Specification

Dr. Ulrich Schade
Dr. Michael R. Hieb
Dr. Miłosław Frey

By this document, a specification of a BML grammar is presented. BML (Battle Management Language) has been envisioned at beginning of the millennium as language that can be used to give orders to simulated units. The following short elucidation about the development of BML is taken from Pullen, Carey et al. (2008a).
[…]

## 1 The Linguistics Principles behind C2LG

In linguistics, a (formal) language is defined by a (formal) grammar (Chomsky, 1957). In short, a (formal) grammar is a quadruple (cf. Partee, ter Meulen & Wall, 1990, pp. 437ff.). It consists of a set of so-called "terminal symbols", a set of "non-terminal symbols", a starting symbol that is part of the set of non-terminals, and a set of production rules. The terminal symbols are the words of the language in question. Thus, the set of terminal symbols is nothing less than the language's lexicon. The non-terminal symbols represent meaningful expressions and thus sequences of lexical items (words), and the rules define how these lexical elements can be combined.

[…]

## 2 Remarks on Notation

In order to be usable as a specification, this document is assumed to be rather strict with notation. C2LG as grammar uses formal context-free grammar rules for building constituents. We will present the rules notated in type "Arial" in the usual form in which the left side and the right side of the rule is separated by an arrow ("→") that means "is expanded to". As the rules are of the context free type, the left side of each rule consists of exactly one non-terminal symbol whereas the right side is a sequence of non-terminal and terminal symbols. Non-terminal symbols start with a capital letter. Terminal symbols (words) are always given in the italic type. Most terminal symbols are taken from the value lists from respective attributes of the JC3IEDM. These are written in normal letters which includes the first letter. However, for some of JC3IEDM's attributes not the values are used as words but the so called physical values (which are much shorter). These physical values consists of capital letter only. But, of course, if used as terminal symbols in C2LG rules they are written in the italic type. Some terminal symbols are not taken from the JC3IEDM. These are so called keywords that are used in the C2LG in order to identify the thematic role of the constituent they start in an un-ambiguous way. These keywords start consist of normal letters, are given in the italic type but are also typed bold. Elements that are part of the right side of the rule but are optional are set into round brackets.

Normally, rules are provided separate from explaining text. If parts of rules are cited in the explaining text they are enclosed by quotation marks. Besides, the notation conventions listed above also are valid.

Examples of C2LG expressions are sequences of terminal symbols. Thus, they are written in the italic type. Those parts of such expressions that are of special interest in the respective section are highlighted in blue.

# 3 Orders, Requests, and Reports in C2LG

In this Chapter, we will described how orders, requests, and reports and expressed in C2LG in general. That means, we will discuss the appearance of general skeleton of these forms of military communication. Later, we will discuss C2LG constituents that are used as building blocks to fill the skeletons. All forms of military communication – that means all orders, requests, and reports – should have a header that has to include at least the sender, the addressee, and the point in time when the communicatory contribution is sent. Other information, e.g., a security classification, might be added. With this in mind, the following rules hold:

Rules 3.0.a) to 3.0.d)
3.0.a) Order → Header OrderBody
3.0.b) Request → Header RequestBody
3.0.c) Report → Header ReportBody

3.0.d) Header → Sender Addressee SendingTime (SecurityClassification)

"Sender" and "Addressee" are to be expanded by the name of the respective unit, "SendingTime" is to be expanded by a datetime value (cf. section 4.4), and "SecurityClassification" is to be expanded by a value from the list of the security values that are valid for the operation in question. E.g., "*NATO Unclassified*" or "*NATO Secret*" would be valid values in a NATO led operation.

## *3.1 Orders*

We will start our discussion of the general skeleton of C2LG communication with the discussion how to build orders, or to be more precise, the body of an order. Most of what is discussed in this section already had been lain down in Schade & Hieb (2006a, b).
[…]

## *3.2 Requests*

From the perspective of speech act theory (Searl, 1979), orders and requests both are directives. The purpose of a directive is to get the addressee to execute an action as intended by the speaker (or the sender). The difference between order and request lies in the right the

sender has to demand the execution of the action in question. With respect to an order, the sender is superior and can expect rightfully that the addressee will execute the tasks that are assigned to him by the sender. In contrast, with respect to a request, the sender is not superior to the addressee and thus cannot expect that requested tasks are executed.
[…]

### *3.3 Reports*

Reports differ from orders and requests. […]

## 4 The Building Blocks

In chapter 3, we discussed how orders, requests, and reports are expressed in C2LG. Since the focus of chapter 3 is the sentence level, we discussed what kind of sentence-like expressions can be used. In addition, we showed how these sentences can be constructed as sequences of constituents. In this chapter, we will focus on these constituents. We will explain how to build the constituents as sequences of lexical items and provide examples to illustrate the discussion. In these examples, we will mark the constituent in question in blue, but provide whole sentence-like expressions or even expressions that consists of many sentences. Showing the constituents in such a large context, we hope to show the interconnections among the constituents.

Constituents, as has already been commented in section 1.1, are the linguistic specifications of the 5 Ws. They express the thematic roles even more exactly than the 5Ws and therefore are the natural building blocks of sentence-like expressions from military communication. The building blocks are

a) verbs, mostly task verbs and event verbs, that specify the “What”, (discussed in section 4.1),
b) denotations for organizations, specifying the “Who”, (discussed in section 4.2),
c) constituents describing spatial aspects, the “Where”, (discussed in section 4.3),
d) constituents describing temporal aspects, the “When”, (discussed in section 4.4),
e) modifier constituents, the “How”, (discussed in section 4.5),
f) constituents to express the purpose, the “Why”, (discussed in section 4.6), and
g) constituents that in the case of reports express aspects of certainty (discussed in section 4.7).

### *4.1 Actions and Events – The “What”*

As has already been said, verbs are the central elements, linguists would say the “head”, of a sentence. In accordance to this, tasking verbs are the central elements of single task assignments and single tasking report lines. In addition, event verbs are the central elements of single lines from event reports. All these single lines, the sentence-like expressions of C2LG, are shaped by their verbs’ frames. Tasking verbs and event verbs especially determine whether there is an “Affected”. In the 5W-terminology, a verb is the “What”. More often than not, what we call “Affected” – the verb’s direct object in a natural language expression – is also considered part of that “What”. This is a misconception because “Affected” is a thematic

role. It is a “Whom” instead of a “What”. The sequence of lexical items that fills the “Affected” role forms a constituent by itself and has to be looked at as unique block. Since this constituent often denotes an organization, e.g., a unit that is affected by tasks like ambush, attack, block etc., “Affected” often is structured like a “Who”. Thus, we will discuss it in the next section in greater detail.
[…]

## *4.2 Denoting Organizations – “The Who”*

In the task assignments, Organizations are denoted as “Tasker”, the superior organization whose commander issues the order, as “Taskee”, the organization that is ordered to execute the task, or as “Affected”, the organization that is affected by the task. In all cases, the organization is referred to by its unique name under which it can be unambiguously found in the underlying data base. In reports, organizations are denoted as “Executer” or, again, as “Affected”. As has already been mentioned, the reporter does not always know the name of an organization mentioned in the report. Thus, in reports, organizations can be referred to not only by their names as will be explained later in this section.
[…]

## *4.3 Expressing Spatial Aspects*

In this section, we will specify how the “Where” for single task assignments (orders) and single report lines is expressed in CBML. To be more precise, we will specify how C2LG expresses spatial information there. This includes representation for the following thematic roles: origin, path, destination, direction, and location. In principle, in C2LG, there are two kind of constituents that represent spatial information connected to assigned and reported tasks. The non-terminals associated with these constituents are called “AtWhere” and “RouteWhere”. As C2LG is a “lexical grammar”, it depends on the task verb which of them is to be used. In general, task verbs that denote (among other things) a movement go along with “RouteWhere” whereas task verbs that do not include a moving demand an “AtWhere”. “AtWhere” is also used in single reports that do not report about tasks (event reports and status reports including position reports).
[…]

## *4.4 Expressing Temporal Aspects*

In this section it will specified how the “When” for single task orders and single reports is expressed in CBML. We have to differentiate between temporal constituents that refer to points in time expressed by denoting the date time and temporal constituents that connect the temporal aspects of an action (the target action or subject action) to specific temporal specifications of another action (the relatum action or object action), e.g. stating that the target action should start as soon as the relatum action ends. The former temporal constituents are called “specific temporal constituents” and the latter one “conditional temporal constituents”.

### *4.4.1 C2LG Rules for Specific Temporal Constituents*

In single task orders, a task is assigned to a unit. In accordance with the JC3IEDM, it can be specified by C2LG expressions when this task has to start and when it has to end. To specify when the assigned task should start, "StartWhen" is used. To specify when the task should end, "EndWhen" is used.

<u>Rules 4.4.a) and 4.4.b)</u>

4.4.a) StartWhen → ***start*** TemporalModifier DateTimeValue
4.4.b) EndWhen → ***end*** TemporalModifier DateTimeValue

Both expressions consists of three parts. The first part is a keyword which is "***start***" in the case of "StartWhen" and "***end***" in the case of "EndWhen". The second is a temporal modifier ("TemporalModifier") and the third is a date time expression ("DateTimeValue"). Temporal modifiers are for example "*at*", "*not later than*", and "*as soon as possible but not later than*". The list of temporal modifiers consists of values of JC3IEDM's attribute "action-task-start-qualifier-code". (JC3IEDM's attribute "action-task-end-qualifier-code" provides the same values.) The current list for C2LG is provided by rule 4.4.c).

<u>Rule 4.4.c)</u>

TemporalModifier → {*AFT, ASAP, ASAPAF, ASAPNL, AT, BEF, NLT, NOB*}

The meanings (values) of these physical values are as follows: *AFT* = *after*, *ASAP* = *as soon as possible*, *ASAPAF* = *as soon as possible after*, *ASAPNL* = *as soon as possible but not later than*, *AT* = *at*, *BEF* = *before*, *NLT* = *not later than*, *NOB* = *not before*.

There are even more values for the JC3IEDM attributes, but those either denote lack of knowledge (*UNK* = *unknown*, *TBD* = *to be determined*) or do not come together with a specific denotation of a date time (*ONCALL* = *on call*, *ONCDWD* = *on codeword*, INDEF = *indefinite*, *UNTFRN* = *until further notice* / the latter two are values only for "action-task-end-qualifier-code"). All these values are currently not part of the C2LG specification.

"DateTimeValue" either is specified as a specific date time or it is "*now*". In the case, the default value "*now*" is used, the date time is identified to the sending time of the respective message (order, request, report). The format used for denoting date times is not further specified in C2LG. However, it is highly recommended to specify this format if C2LG should be used for C2 communications. Otherwise misunderstandings might result. It is also allowed to expand "DateTimeValue" with the unique name for a specific point in time that had been put down in the underlying common data base. Normally, such names refer to temporal start or end points of phases defined for the respective operation.

In single reports, "When" is used to provide temporal information. "When" is expanded as follows.

<u>Rules 4.4.d) to 4.4.f)</u>

4.4.d) When → ***start*** *at* DateTimeValue
4.4.e) When → ***ongoing*** *at* DateTimeValue
4.4.f) When → ***end*** *at* DateTimeValue

The construction of these constituents corresponds to those of "StartWhen" and "EndWhen" above. Again, there is a keyword, a temporal modifier and a date time value. There are however, two minor differences: First, in the current C2LG version, for single reports only "*at*" is used as modifier. Second, for reports, in addition to "***start***" and "***end***", also "***ongoing***" is used as keyword. Temporal constituent always start with one of these keywords. Thus, they can be recognized by them.

### *4.4.2 Examples of Specific Temporal Constituents*

The following examples are taken from our papers on C2LG. They show complete single task orders or complete single reports in order to provide the temporal constituents in their contexts. The temporal constituents themselves have been highlighted in blue.

Example 4.4.a)

The leader of the Multinational Division "West" (lead by a Spanish element) orders the 13th Dutch Mechanized Brigade (M_BDE13(NL)) to march along a pre-defined route labelled "*DUCK*" during the first phase (lasting from point in time "*Phase1A*" to point in time "*Phase1E*") of an operation.

```
[task assignment] march    MND-West(SP) M_BDE13(NL) along  DUCK
                           start  at  Phase1A    end before Phase1E    label_3_11;
```

Example 4.4.b)

The leader of the 2nd company reports to her battalion leader:
"Arrived at Control Point 3; start attacking snipers."
(The group of snipers had been already labelled "*label-en-1*" in the communication before the report above was made. Therefore, this label is used as "Affected" in the task report. )

```
[position report] own position Coy2 at ControlPoint3 start at now fact label-rpos4;
[task report] attack Coy2 label-en-1 at ControlPoint3 start at now fact label-rtsk4;
```

### *4.4.3 C2LG Rules for Conditional Temporal Constituents*

In single task orders the temporal constituents can be expressed with reference to another task or an event. The assigned task is called "target task" or "subject task", and the action (task or event) referred to is called "object action" or better "relatum action".

In order to do this, "StartWhen" and "EndWhen" are extended not by the rule given above in section 4.4.1 under "*C2LG Rules for Specific Temporal Constituents*" (rules 4.4.a) and 4.4.b)) but by the following ones (rules 4.4.h) and 4.4.i)):

Rules 4.4.h) and 4.4.i)

4.4.h) StartWhen → ***start*** TemporalAssociationModifier1 RelatumAction
4.4.i) EndWhen → ***end*** TemporalAssociationModifier2 RelatumAction

The keywords in these rules are the same as above. The temporal modifier is substituted by a temporal association modifier that specifies the association by which the target task is related to the relatum action. Instead to a point in time, this rule refers to the relatum action ("RelatumAction"). The reference is carried out by the use of a unique label that must have been assigned to the relatum action in a preceding C2LG expression.

The following list is a list of the values C2LG allows as temporal association modifiers. They are taken or inspired from the list of physical values designed for JC3IEDM's attribute "action-temporal-association-category-code". However, for our purposes the list of these physical values from the JC3IEDM is incomplete, so it had be enhanced to what is proposed by rules 4.4.j) and 4.4.k).

Rules 4.4.h) and 4.4.i)
4.4.j) TemporalAssociationModifier1 → {*STRBEF, STRSTR, STRDUR, STREND, STRAFT*}
4.4.k) TemporalAssociationModifier2→{*ENDBEF, ENDSTR, ENDDUR, ENDEND, ENDAFT*}

The meanings of these values are as follows:

| | | |
|---|---|---|
| *STRBEF* | = | target action starts before relatum action starts |
| *STRSTR* | = | target action starts when relatum action starts |
| *STRDUR* | = | target action starts during relatum action |
| *STREND* | = | target action starts when relatum action ends |
| *STRAFT* | = | target action starts after relatum action ends |

| | | |
|---|---|---|
| *ENDBEF* | = | target action ends before relatum action starts |
| *ENDSTR* | = | target action ends when relatum action starts |
| *ENDDUR* | = | target action ends during relatum action |
| *ENDEND* | = | target action ends when relatum action ends |
| *ENDAFT* | = | target action ends after relatum action ends |

### *4.4.4 An Example of a Conditional Temporal Constituent*

The following shows an example in which a conditional temporal constituent is used. Naturally, in the example, there two task assignments such that the temporal constituents of the one task can serve as relatum information in the conditional temporal constituent.

Example 4.4.c)
The leader of the battalion 661 orders his 2nd company to occupy the "Prins Willem-Alexander" bridge at Parnass. This task is labelled by "*label-023*". The temporal constituents for this task are expressed as specific temporal constituents.

Occupying the bridge is the pre-condition for securing it. Consequently, the next task, namely to secure the bridge, should start when the occupation has ended as is ordered by the second task assignment in this example.

[***task assignment***] *occupy BN-661 Coy2 Prins Willem-Alexander Brug at Parnass*
***start** at TP1 **end** before TP2 **in-order-to** enable label-o24 label-o23;*

[***task assignment***] *secure BN-661 Coy2 Prins Willem-Alexander Brug at Parnass*
***start** STREND label-o23 label-o24;*

## *4.5 Expressing Modifiers*

In this section it will specified how the "How" for single task orders is expressed according to the C2LG.

[…]

## 4.6 *Expressing Purpose*

In this section it will specified how the "Why" for single task orders, for single request lines, and for single lines of task reports is expressed according to the C2LG.

### *4.6.1 C2LG Rules for the Purpose Constituent*

The purpose constituent starts with the keyword "***in-order-to***". This keyword is followed by a task influencing verb and the label of a single task order. If the order as whole includes a command intent expression, the label of one basic expressions of this command intent might be used instead of the label of a single task order.

Rules 4.6.a) to 4.6.c)

4.6.a) Why → ***in-order-to*** TaskInfluencingVerb TaskLabel
4.6.b) Why → ***in-order-to*** TaskInfluencingVerb IntentExpressionLabel
4.6.c) TaskInfluencingVerb → {*allow, assist, cause, enable, prevent*}

Task influencing verbs are verbs that have an action or a state as "theme". So, in C2LG one is not allowed to say "attack … enemy … in order to destroy enemy". Instead, the destruction of the enemy has to be expressed as a task by its own. Say, the task to destroy the enemy gets the label "task-d". Then the original attack that should result in the destruction of the enemy is expressed as "attack … enemy … in order to enable task-d". In this manner, tasks can be chained together such that the logic behind the operation is expressed. Consequently, one can not only refer to other tasks but also to parts of the command intent such that the chain of tasks finally results in causing a desired state as expressed in the intent. In general, the task influencing verbs "*allow*", "*assist*", "*enable*", and "*prevent*" are followed by a label referring to a task. Normally this is a task assigned to an own unit in paragraph 3b or paragraph 3c. It also might be a task mentioned as key task in paragraph 3a "Execution – Intent" or a task mentioned in paragraph 1a "Situation – Enemy Forces" or paragraph 1b "Situation – Friendly Forces". Especially the influencing verb "*prevent*" is – of course – often followed by a task whose execution by the enemy has been reported or is anticipated, and is thus mentioned in paragraph 1a. In contrast to the verbs mentioned, the influencing verb "*cause*" normally is followed by a state which is a desired end state from paragraph 3a.

### *4.6.2 Examples of Purpose Constituents*

Example 4.6.a)

Example 4.4.c), as given above, already shows the chaining feasible by the use the Why constituents. In that example that is repeated here as example 4.6.a) the execution of the first assigned task is a preposition of the second: Occupying the bridge enables its securing. Thus, the basic order expression for the occupation task notates the securing task (labelled by "*label-o24*") as its purpose.

[***task assignment***] *occupy BN-661 Coy2 Prins Willem-Alexander Brug at Parnass*
***start*** *at TP1* ***end*** *before TP2* ***in-order-to*** *enable label-o24 label-o23;*

[***task assignment***] *secure BN-661 Coy2 Prins Willem-Alexander Brug at Parnass* *__start__ STREND label-o23 label-o24;*

Example 4.6.b)

Example 4.6.b) is a longer one. It is taken from the preparation of NATO RTO MSG-048's 2009 experiment, but had been simplified in some details for clarification. The Combined Arms Battalion 66 is ordered to attack southeast, taking and securing two objectives (objective DOG and objective LION) and thereby defeating or bypassing enemy forces. The enemy forces are assumed to be already reduced to 40% combat effectiveness. Battalion 661 consists of two armoured squadrons (Coy1 and Coy3), each having 14 main battle tanks, two mechanized companies (Coy2 and Coy4) and a recce platoon. The example shows the orders for one of the armoured squadrons (Coy1) and one of the mechanized infantry companies (Coy3) who are ordered to cooperate in coordination. The sketch for an overlay of this operation is given in figure 1 for better understanding.

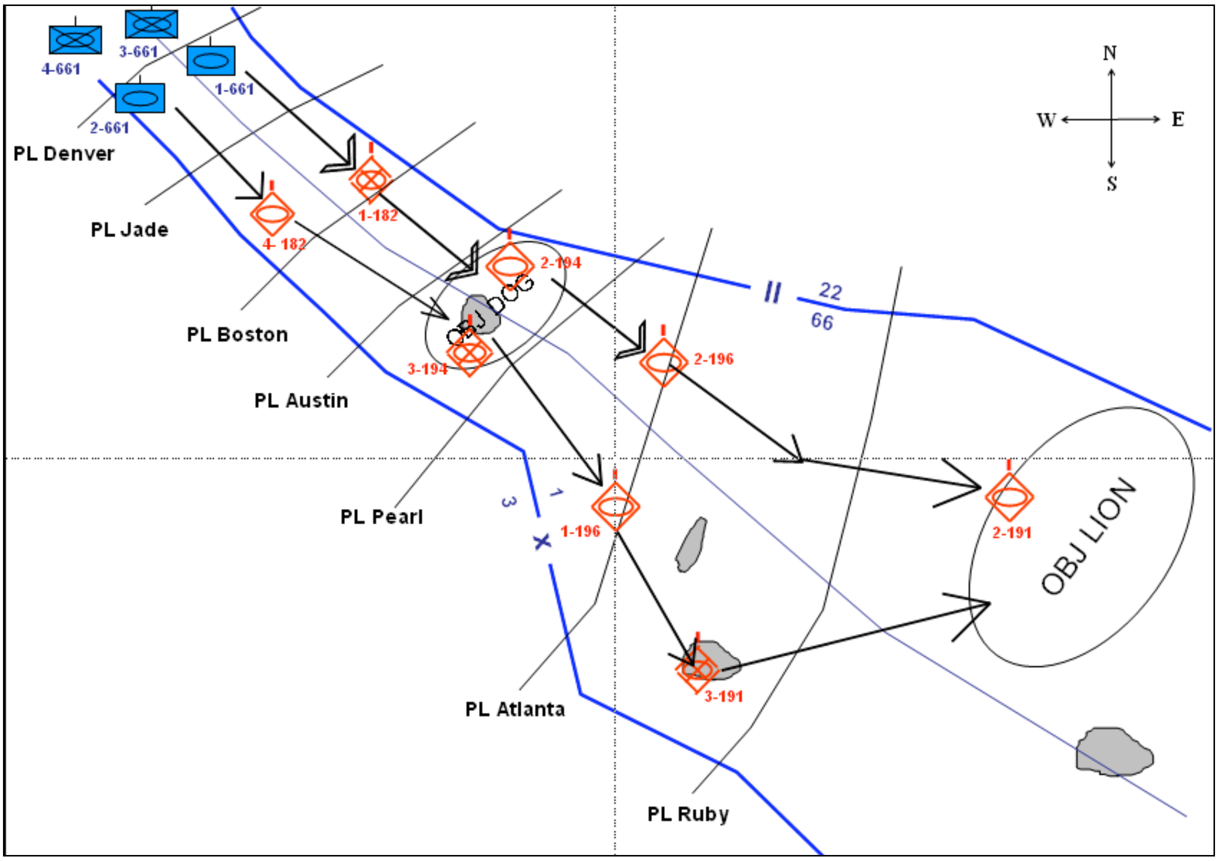

Figure 1: The figure shows the sketch of an overlay for an operation. Part of the tasks assignments of the order by which that operation is ordered is in given in example 4.6.b)

In the example, categorization term are not included and the labels of the sentence-like expressions are coloured in red (whereas the Why-constituent is coloured in blue). The detailed explanation below the example will use the red coloured labels to refer to the respective sentence-like expressions.

Command intent:
*secure BN-661 Lion at Lion __ongoing__ at TP4 RPTFCT label-ci1;*

Tasks assigned to Coy1:

```
deploy BN-661 Coy1 at Denver end before TP0
           in-order-to enable label-o11 label-o10;
advance BN-661 Coy1 from Denver to Boston start at TP0
          in-order-to enable label-o12 label-o11;
fix BN-661 Coy1 Red-1-182 at Boston end nlt TP1
          in-order-to enable label-o33 label-o12;
advance BN-661 Coy1 to Austin start at TP1
          in-order-to enable label-o14  label-o13;
fix BN-661 Coy1 Red-2-194 at Dog end nlt TP2
          in-order-to enable label-o35  label-o14;
advance BN-661 Coy1 to Atlanta start at TP2
          in-order-to enable label-o16  label-o15;
fix BN-661 Coy1 Red-2-196 at Atlanta end nlt TP3
          in-order-to enable label-o37 label-o16;
advance BN-661 Coy1 to Ruby start at TP3
          in-order-to enable label-o18  label-o17;
fix BN-661 Coy1 Red-2-191 at Lion end nlt TP4
          in-order-to enable label-o39  label-o18;
seize BN-661 Coy1 Lion at Lion end nlt TP4
          in-order-to cause label-ci1  label-o19;
```

Tasks assigned to Coy3:

```
deploy BN-661 Coy3 at Denver end before TP0
          in-order-to enable label-o32 label-o30;
support BN-661 Coy3 Coy1 at Troy start at TP0 end at TP4 label-031;
attspt BN-661 Coy3 Red-1-182 from Denver to Boston start at TP0 end nlt TP1
          in-order-to enable label-o12  label-o32;
destroy BN-661 Coy3 Red-1-182 at Boston end nlt TP1
          in-order-to enable label-o13  label-o33;
attspt BN-661 Coy3 Red-2-194 from Boston to Dog start at TP1 end nlt TP2
          in-order-to enable label-o14  label-o34;
destroy BN-661 Coy3 Red-2-194 at Dog end nlt TP2
          in-order-to enable label-o15  label-o35;
attspt BN-661 Coy3 Red-2-196 from Dog to Atlanta start at TP2 end nlt TP3
          in-order-to enable label-o16  label-o36;
destroy BN-661 Coy3 Red-2-196 at Atlanta end nlt TP3
          in-order-to enable label-o17  label-o37;
attspt BN-661 Coy3 Red-2-191 from Atlanta to Lion start at TP3 end nlt TP4
          in-order-to enable label-o18  label-o38;
destroy BN-661 Coy3 Red-2-191 at Lion end nlt TP3
          in-order-to enable label-o19  label-o39;
```

The operation is divided into four phases (from TP0 to TP1, from TP1 to TP2, from TP2 to TP3, and from TP3 to TP4). The points in time, labelled TP0 to TP4, respectively, are specified by exact date times in the underlying data base.

The first line of the example, labelled by "*label-ci1*", is part of the command intent. It says that at the end of the ordered operation, at TP4, the Combined Arms Battalion should be in a position such that it is securing objective Lion. The last basic order given to Coy1, labelled by "*label-o19*" has the purpose to cause that this state comes true.

The first basic orders to Coy1 and Coy3 (*label-o10*, *label-o30*) are about the deployment of the units at line of departure "*Denver*". Departure is the precondition of starting the operation. Thus departure enables the first moving actions towards "*Lion*", the advance (in strength) of

Coy1 to phase line "*Boston*" (*label-o11*) and Coy3's supporting attack (*label-o32*) in coordination with Coy1's fixing the enemy unit "*Red-1-182*" (*label-o12*). The advance of Coy1 and the supporting attack of Coy3, both enable the fixing of the enemy ("*Red-1-1-182*").

The second order for Coy3 (*label-o31*) is a more general one, saying that Coy3 should support Coy1 during the whole operation.

Fixing the enemy by Coy1 allows its destruction by Coy3 (*label-o33*). This, on the other hand enables the advance of Coy1 to phase line "*Austin*" (*label-o13*). From there, enemy unit "*Red-2-194*" can be fixed at objective Dog (*label-o14*). Again, the fixing is supported by Coy3 (*label-o34*). The fixing allows the destruction of the enemy at Dog by Coy3 (*label-o35*). And, again, this enables the further advancing of Coy1 to phase line "*Atlanta*" (*label-o15*). There, another enemy unit ("*Red-2-196*") is fixed by Coy1 (*label-o16*) under support of Coy3 (*label-o36*) allowing the destruction of "*Red-2-1-196*" by Coy3 (*label-o37*). This enables Coy1 to advance to phase line "*Ruby*" (*label-o17*) and start the attack of objective Lion by fixing enemy unit "*Red-2-191*" there (*label-o18*). Coy3 supports the attack (*label-o38*). The fixing of the enemy in the end allows its destruction by Coy3 (*label-039*) which enables Coy1 to seize the objective (*label-019*). As already had been said, this allows securing the objective such that the desired state as formulated in the intent is reached.

### *4.7 Expressing Certainty*

In this last section of the "building block" chapter, we will give the C2LG rules for expressing certainty (section 4.7.1) in addition to some examples (section 4.7.2).
[…]

# 5 Command Intent

Command and Control is art. There are many unforeseeable aspects that can influence an operation. Explicit and detailed orders quickly become outdated and thus, following them by letter is the surest way into disaster. The answer to this problem is adaptability. Under the head words "Network Centric Warfare" Alberts and Hayes discussed "adaption" as "the ability to alter force organization and work processes as the situation and/or environment changes" (2003, p. 153). After the Napoleonic Wars, the Prussian Army established adaptability and agility by "Mission Command", a command style that matches the necessities of network centric operations: "information age […] creates conditions where such a command philosophy [mission command] is the essential bedrock for success" (Storr, 2003, p. 93). It has to be mentioned that this command style had been at least occasionally used through all periods of history. E.g., Alberts and Hayes refer to Nelson's tactic at the Battle of Trafalgar (1805) as their prime example to illustrate their reasoning in favour of agility (2003, pp. 28ff.). Traditionally, however, Mission Command is attributed to Prussia's supreme commander, Helmuth Karl Bernhard Graf von Moltke (Moltke the Elder) (1800-1891) who established mission command doctrinally (Regling, 1983, p. 384; Borgert, 1983, p. 427f.).

How command intent is represented in C2LG has been discussed in detail in Hieb & Schade (2007). In this chapter, we will summarize and actualize the most important aspect given there. Albert and Hayes discuss intent in "Understanding Command and Control" and distinguish between intent, Command Intent, and Commander's Intent (Albert & Hayes, 2006, p. 38). Commander's Intent implies a single individual in command, while Command Intent is a newer term that implies a group or collective making decisions. The term intent is more general yet. With respect to C2LG, we prefer (and thus use) "Command Intent". It is in our opinion the best term for discussing transitioning from a well-established command process relying upon written and verbal communication to a more flexible command environment, that can still support the same processes, but is built upon a formal grammar.

According to Wade (2005, p. 1-22), who uses the classical term, commander's intent consists of three components, namely the desired *End State*, summing up the conditions under which a mission is accomplished, a list of *Key Tasks*, those tasks that the commander thinks to essentially must met to achieve the desired end state, and in the context of the broader operational context of the mission, the *Expanded Purpose*. Following this listing, we set the following rule for the expansion of command intent ("*CI*"):

Rules 5.a)
CI → (KeyTasks) EndState (ExpandedPurpose)

With respect to commanding simulated units, only the desired end state is of relevance. The broader picture often will not be part of the simulation, therefore it will not help to express a expanded purpose for a simulation system. The key tasks would also be "normal" tasks assigned to some units. Thus, they are already given to the simulation system in question, and emphasising them only directs a human's attention to them but does nothing for a system. The desired end state however can be exploited for the simulation if a planning system is added to the architecture that is able to transform the desired states to plans (cf. Nazih & Schade, 2009). Because of these reasons, in the rule given above, "ExpandedPurpose" as well as "KeyTasks" are optional whereas "EndState" is mandatory.

As "EndState" represents the desired situation that can be reported if the mission is successfully accomplished, "EndState" can be represented by a sequence of basic report expressions. For these report expressions holds that their temporal constituent refer to a point of time in the future, when the desired situation has to be achieved, besides the certainty constituent is set to "*reported as fact*", saying that at that time of the future the desired state can be reported as fact. Key Tasks are, of course, tasks. Thus "KeyTasks" can be formulated as both basic order expressions. Expanded Purpose is similar to the End State, but expresses more general aspects of the resulting situation. In short, the End State is about the resulting situation form the military perspective whereas the Expanded Purpose also considers other, e.g., political, consequences and results. Being the description of a state, "ExpandedPurpose" like "EndState" is expanded as a sequence of basic report expressions, but these will include more reports of event type that refer to political situations, such as "*POW return*" or "*peace conference*". The rules for the parts of command intent therefore are

Rules 5.b) to 5.d)
5.b) KeyTasks → OB*
5.c) End State → RB*
5.d) Expanded-Purpose → RB*

With respect to "KeyTasks" there is an additional feature that has to be considered. In a normal task assignment as expressed in paragraph 3b) to 3c) of the operation order, tasks are assigned directly to a subordinate unit. Thus, tasker and taskee are well known and can be referred to by name. In a key task as expressed in a command intent, however, the commander might order his subordinate to assign a task, the key task in question, to one of his units. In this case, the commander does not specify which unit should execute the task. In contrary, the subordinate commander is assumed to chose that one of his units that is the most appropriate one. In other words, while expressing the key task, the commander cannot refer to the taskee by name. For this, C2LG uses the replacement term "*OPEN*" that allowed to expand "Taskee" in any key task expression of a command intent. The following example illustrates the use of this replacement term.

Example 5.a)
Example is based on example 4.6.b). That example shows parts of an order by which the commander of battalion BN-661 commands an operation that finally ends with seizing and securing of objective "Lion". In his own command intent, the commander of brigade BDE-66, superior to the commander of the battalion BN-661, could have demanded the seizing of "Lion" as a key task of the brigades broader operation by commanding the commander of BN-661 to assign that task to one of his companies, e.g. in order to enable another crucial task.

*[command intent – keytasks] seize BN-661 OPEN Lion at Lion* ***end*** *nlt TP4*
***in-order-to*** *enable label-ci13 label-ci12;*

# 6 Additional Aspects

In this final chapter, some aspects are addressed that deal with more general BML-related questions. In the first section of the chapter (section 6.1), we discuss the role the data model – or to be more precise, the JC3IEDM – can and should play in military communication in general, and in giving orders by using a version of BML in particular. In the second section (section 6.2), we will light up the relationship between C2LG and XML schemata that are based on the grammar.

## *6.1 The Role of the Data Model*

The task of a data model like the JC3IEDM is to determine the rules and the formats by which data can be exchanged among different organizations such that all the organizations have the data which are relevant for that organization in an appropriate and well understood format. If there is military communication, this communication is information by itself and may therefore have to be made available for other (partner) organizations that have not been part of the communication themselves. Thus, from JC3IEDM's point of view, these communications have to be stored in the database as data and this data has to be exchanged among the organizations in question. With respect to plans and orders, the JC3IEDM goes two ways. The parts of which an order consists (for example, Word documents for each part of a five paragraph operation order) can be stored and exchanged as Word documents. This is helpful for the data exchange task as described but does not break down the orders to a level at which they can be analyzed automatically, e.g. by a simulation system. The second way JC3IEDM can represent orders is to represent the plan behind them by representing all of the to-be-assigned tasks (using table "action-task" and connect the involved object representation as required) and representing as well the interconnections of these tasks by the tables "action-functional-association" and "action-temporal-association". If, then, for a represented plan, the "action-task-category-values" for all the assigned tasks involved are set from "plan" (PLAN) to "order" (ORD), the database represents the fact that the respective plan has been ordered.

If the latter representation is chosen, the database does not contain the order as an order but "only" the information that some actions have been ordered. As previously said, orders are directives. They are given in order to make the addressee execute the tasks assigned to him.

Orders initiate actions, whereas information about tasks that are ordered does not. To illustrate this point, consider the following two fictional descriptions of ordering and the results achieved by the orders. The first is from ancient times, from the Iliad, the second is from modern SF.

„Laut erscholl sein durchdringender Ruf in die Schaaren der Troer:
Auf, ihr reisigen Troer, hinan! Durchbrecht der Argeier
Mauer, und werft in die Schiffe die schreckliche Flamme des Feuers!
Also ermahnte der Held; und Aller Ohren vernahmen's.
Grad' andrang zu der Mauer die Heerschaar; Jene begierig
Klommen empor die Zinnen, geschärfte Speer' in den Händen.“
(Homer's *Iliad*, as translated into German by Heinrich Voß, 1882, p. 188, 12th song, verse 439 to verse 444)

““Helm, roll ninety degrees port!”
No one on that wounded, half-broken bridge, and Honor least of all, perhaps, recognized the cool, sharp soprano which cut cleanly through the chaos, but the helmsman clinging to his own sanity with his fingernails recognized the incisive bite of command.
[…]
“Port broadside stand by for Fire Plan Delta Seven,” that soprano rapier commanded, and confirmations raced back from *War Maiden*'s undamaged broadside […].”
(Weber, 2004, pp. 151f.)

Obviously, there is no relevant difference between the examples. In both cases the addressees act as required, and they act without any inquiry or discussion. On the contrary, they anticipate the orders eagerly and act with raised spirit.

From the linguistic point of view, speakers (senders) produce verbal expressions to convey specific communicative intentions. With respect to directives in general and orders in particular, a speaker (sender, tasker) wants to get the addressee to do something. This intention is called the *illocutionary force* of a speech act (Austin, 1962; Dörge, 2006). With respect to orders, the sender can rightfully assume that his speech act will be successful, which means that the addressee indeed acts as intended by the speaker. The right to assume that the addressee will execute the order is anchored in the military hierarchy, in every military doctrine and in the self-conception of military organizations. It is also the linguistic core of the speech act of ordering.

It has been suggested that instead of using a BML to express an order and then to send the BML order to the taskee, the order should be represented as a plan in a JC3IEDM database. After that the “action-task-category-code” values of all included tasks could be changed from “plan” (PLAN) to “order” (ORD), and the representation and the change could be distributed to all other C2 systems or simulation systems with the assumption that every organization or simulation system would interpret the distributed data as orders upon which it must act. This suggestion wilfully ignores the difference between an order and its illocutionary force and the information about the order as stored in the database (which as information does not have the illocutionary force of a directive). This suggestion has to be refused emphatically.

## 6.2 *Grammar and XML Schemata*

Once a grammar has been defined, it can easily transformed into an XML schema. In order to provide an example, let us take a look at the "RouteWhere" rules for spatial constituents as given in section 4.3.1 (rules 4.3.e) to 4.3.g) and 4.3.c)).

Rules 4.3.e) to 4.3.g) and 4.3.c)
4.3.e) RouteWhere → ***along*** RouteName
4.3.f) RouteWhere → ***towards*** Location | ***towards*** Bearing
4.3.g) RouteWhere → (***from*** Location) ***to*** Location (***via*** Location*)

4.3.c) Location → LocationName | Coord

A possible transformation of these rules into a schema is shown in Figure 2.

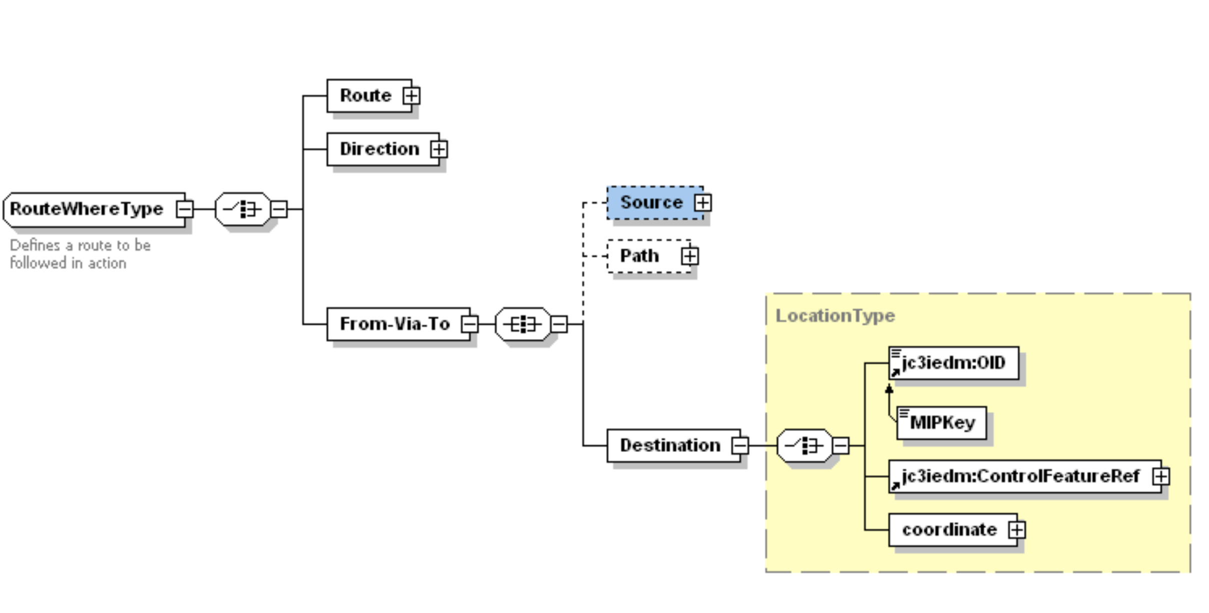

Figure 2: The part of an XML schema that can be developed "directly" out of C2LG's rules for "RouteWhere"

In order to interpret the schema, the following has to be taken notice of. First, the schema uses a so called "LocationType" determining that a location is either the name of a JC3IEDM object (JC3IEDM:OID) or a coordinate. This responds to the "Location" rule above although it could be restricted even more by determining that the JC3IEDM object in question has to be of type facility or feature. Second, the keywords that are used in C2LG will be used to identify the corresponding schema elements. Thus, a constituent starting with keyword "***along***" will use the "Route" element in the schema, a constituent starting with "***towards***" the "Direction" element, a constituent with "***from***" the "Source" element, a constituent with "***via***" the "Path" element, and a constituent with "***to***" the "Destination" element. As determined by the third "RouteWhere" rule above, the latter three are part of the "From-Via-To" element (no better label had been found) under the additional condition that the "Source" element and the "Via" element are optional whereas the "Destination" element is mandatory.

C2LG's rules have been transformed into a schema under the aegis of NATO RTO MSG-048 "Coalition Battle Management Language" (cf. Pullen, Hieb et al., 2008). However, the resulting schemata (the IBML schema and its predecessor the JBML schema) are supposed to be used by quite a range of organizations for the exchange of orders and reports among systems. Thus, other stakeholders also influenced how the schema should look like and what structure it should have to have their own needs supported. Thus, although the C2LG's rules have shaped the IBML schema by large there are deviations. Most importantly, in the IBML schema the task verbs do not come with their own sub-schemata. Instead, the rule form for basic order lines (cf. section 3.1) has been transformed into the schema. Thus, it is possible to represent a task assignment in the schema that includes a "RouteWhere" although its "TaskingVerb" – according to the C2LG and according to normal semantic constriction – demands an "AtWhere" (and vice versa). This is not a problem as long as orders or reports are put into the sender system by using a C2LG-GUI – that allows only those inputs that respect the grammar rules – or as long as the input is correct independent of any check. Then the input is transformed into a schema representation that is valid not only according to the schema but also according to the grammar.

# 7 References

[...]