Developing an ATC Grammar using the Review of the Cushing Grammar

in the project seminar

Computational Natural Language Systems

RVS, Faculty of Technology, University of Bielefeld

Report RVS–Occ–01–03

Martin Ellermann*   Mirco Hilbert†

June 28, 2001

Contents

* `mellerma@TechFak.Uni-Bielefeld.de`

† `mail@Mirco-Hilbert.de`

1 Preface

After we analysed the "AIR System Controller Grammar" by Steven Cushing [Cush1994] and described the errors and problems, that came up during this analysis (for details see [Elle2001]), we used these results to develop a new grammar, which suits our purpose to built a parsing software for well-defined ATC/pilot communications.

The grammar described in this document is available from the authors or the RVS Group, on request.

2 Requirements for the ATC Grammar

For the new grammar we decided on the following requirements:

EBNF-near syntax

The grammar syntax should orientate according to the EBNF[1] common in computer science and computational linguistics. That makes the grammar easier to understand for most readers and easily adaptable for other purposes.

Machine processability

The grammar should be machine processable or it should at least be automatically convertable to a form which makes it possible to automatically parse ATC input with it.

Human processability

The grammar should be well structured, clearly arranged and easy understandable for a human reader. That entails that the grammar should be modular for all the usual reasons.

The grammar rules should be as short as possible and as simple as possible. They should be thematically ordered and commented.

Approximation to the ATC standard

The grammar should describe a part of the ATC language and should approximately conform with the ATC standard of the United States (see therefore [ATC] and [AIM]).

Completeness

The grammar should be complete and self-contained, i. e. every non-terminal symbol (for instance <digit_seq>), that is used in any of the grammar rules, must be defined anywhere in the grammar and should be replacable through the grammar rules with a string of terminal symbols (for instance 12345) by a recursive substitution process.

[1] Extended Backus Naur Form, for a specification see for example [Schö1995].

Consequently, every grammar rule should specify (via the recursive substitution) a set of strings that only consist of terminal symbols.

Correctness

The grammar should be as correct as possible, i. e. no correct sentence which belongs to the part of the ATC language that the grammar describes should be rejected and as few as possible incorrect sentences should be inferable.

3 How to fulfil the requirements

3.1 Changing the grammar syntax to an EBNF-near syntax

Some rules of the Cushing Grammar are structured by semantic meta information like "*or*" or "*If . . . else*". This EBNF-nonconforming meta information must be replaced by new meta symbols and the information that they imply must be included in the new grammar rule.

Example

Cushing's rule for an altitude specification is structured by a meta text that implies information about the non-terminal <digit++> which is used in the rule.

<altitude> *If the number of feet is less than 18,000:*
ALTITUDE <digit++> THOUSAND
[<digit> HUNDRED]
else:
FLIGHT LEVEL <digit++>

To correct this and make the rules simpler we divided the rule into two extra rules, one for low altitude less than 18,000 feet and the other one for high altitude 18,000 feet upwards. The number constraints for the non-terminal <digit++> we included directly in the syntactic definition of the altitude specification.

```
# Altitude
# note: <altitude> definition devided into low and high altitude
<altitude>          <altitude_low> | <altitude_high>

# Altitude (if the number of feet is less than 18,000 =>  from 0 to 17,900)
# see:  AIM 4-2-9a
# note: 1,000 feet = 304.80 m
#       <digit> and <digit++> put in concrete terms
<altitude_low>      ALTITUDE (
```

```
                                 <digit> |
                                 <1> (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                              ) THOUSAND [<digit> HUNDRED]

     # Altitude (else =>  from 180 to 450)
     # see:  AIM 4-2-9b
     # note: FL 180 = 18,000 feet = 5,486.40 m
     #       <digit++> put in concrete terms
     <altitude_high>      FLIGHT LEVEL (
                                 <1> (<8> | <9>) <digit> |
                                 (<2> | <3>) <digit> <digit> |
                                 <4> (<1> | <2> | <3> | <4>) <digit> |
                                 <4> <5> <0>
                              )
```

Another consideration when changing the Cushing Grammar to an EBNF-near syntax is that Cushing used some meta symbols that are not defined in the EBNF syntax. For example he used a slash to separate alternative words, a vertical line to separate alternative groups of words and the meta-word "*or*" to separate alternative entire sentences. But only the vertical line is defined in EBNF. So the EBNF-nonconforming symbols must be replaced in this case by vertical lines and some parentheses if necessary.

Cushing also uses non-terminal symbols that includes meta characters. These implied rule information must be replaced by explicid rules in a EBNF-near syntax.

Example

Cushing used the non-terminals <digit++> and for instance <digit+2>. <digit++> should stand for one or more repetitions of <digit> and <digit+2> for exact two repetitions of <digit>.

<digit+i> we consequently replaced by the specified number of i+1 single `<digit>` symbols. And instead of <digit++> we defined a non-terminal symbol `<digit_seq>` which name should stand for "sequence of digits".

```
<digit_seq>          <digit> [<digit_seq>]
```

The whole syntax of the new grammar is defined in the appendix A.1.

3.2 Making the grammar machine processable

One aspect of machine processability is that the grammar have a uniform syntax. This requirement is partly fulfilled when the new grammar is in an EBNF-near syntax, as specified in the subsection 3.1.

In addition, we normalized the internal syntax of the non-terminal symbols. In the Cushing Grammar there are some non-terminals written in mixed capital and lower-case letters, and including blanks, hyphens and underscores. Some tokens even end with ++ or $+i$ as mentioned above. In the Cushing Grammar, therefore, sometimes different syntactical spellings of one non-terminal symbol are used at different positions of the grammar. For machine processability this inconsistency is unacceptable; it makes automatic processing impossible. To avoid this problem we normalized the internal syntax of non-terminal symbols so that they may only to consist of non-capital letters, digits and underscores, enclosed in angle brackets.

As we have seen in the first example on page 2, we marked up comments on a grammar rule with a hash (# ...). As well as making the grammar more easily readable for a human, the hash at the beginning of a line can easily be parsed by a computer program that takes the grammar as input. The hash indicates unnecessary information for a compiling tool that can be ignored in a first processing step.

A last non-syntactical aspect is that the new grammar must be complete (as specified in the requirements on page 1 and in the subsection 3.5) so that a parser cannot enter a state in which it does not find a definition for a non-terminal.

3.3 Making the grammar human processable

To fulfil the requirements specified on page 1 and to make the whole grammar modular and easier to understand for a human reader we divided the grammar in two main parts.

Like the Cushing Grammar the first part contains token definitions for often-used data. The second part consists of two different kind of rules: block rules and sentence rules.

A **token** is a non-terminal symbol that combines a set of semantically closely related fixed terms or short phrases, which are often used in ATC phrases, to the generic term belonging to them. Tokens are for example defined for location and airport names, for craft types and for an altitude specification as mentioned in 3.1.

A **block** is a combination of several sentences which thematically belong together and can be uttered as alternatives depending on the actual situation. Cushing often conjoined such sentences with the meta-word "*or*". We broke up such *or*-rules to make the rules shorter and more clearly arranged.

A **sentence** is a short utterance which ends with a dot (period). A sentence can be a question, an instruction or a general proposition. In the sentence rules the tokens are used as macros or abbreviations for a complex data set to make the sentence rules shorter. A block rule refers to one or more sentence rules but a sentence rule can be used in more than one block definition.

Example

The block rule `<block12>` that specifies the report of a wind shear consists of two sentences `<sentence22>` and `<sentence23>`.

```
# Weather
# flightphase: cruise
# category:    weather
<block12>      <sentence22> <sentence23>
```

The sentence rules specify the syntax of the utterance.

```
<sentence22>   WIND SHEAR (
                  REPORTED | ALERT | ALERTS (TWO | SEVERAL | ALL) QUADRANTS
               ).

<sentence23>   CENTERFIELD WIND <direction> AT <speed> (
                  , <direction> BOUNDARY WIND <direction> AT <speed> |
                  VARYING TO <direction> AT <speed>
               ).
```

In them the tokens `<direction>` and `<speed>` are used which are defined in the tokens section of the grammar.

```
# Cardinal points
<direction>         <quad> | <location>

<quad>              NORTH | SOUTH | EAST | WEST

<location>          NORTHEAST | NORTHWEST | SOUTHEAST | SOUTHWEST


# Speed specification (in knots)
# see: AIM 4-2-11
<speed>             <1> KNOT | <number_gt_one> KNOTS
```

All rules function together for the specific block rule `<block12>` but the tokens can be used in other rules, too, and also could the sentences. And because of the modularity each rule is easy maintainable. The definition of `<sentence23>` is independent from the definitions of `<direction>` and `<speed>` assuming their definitions are correct and that they describe what they are intended to.

A last point concerning human readability is that we grouped together thematically-corresponding block and sentence rules. Also, we added comments to the block rules indicating in which flight phases the blocks could be said and to which semantic category they belong. For some token definitions we added a reference to an official manual ([ATC] or [AIM]) where more information about the ATC syntax for this token or some examples for the usage of the token are found.

3.4 Approximating to the ATC standard

To write a new ATC grammar we have not taken only the sentences that we derived from the phrase rules of the Cushing Grammar as a basis, as we described in [Elle2001] and in the subsection 3.6 below. We also looked at some example phrases that are contained in the ATC handbook as well as in the AIM. We compared these "official" examples with our derived examples to determine compatibility and to conclude from the context in which non-defined tokens occur what their values would be. As already mentioned above, we added references to the ATC handbook or to the AIM at these token definitions.

Some times it was difficult to reconcile the "official" examples with the examples of the Cushing Grammar, because the official manuals often only define the syntax of short ATC phrases whereas the Cushing Grammar mostly defines longer constructions of sentences.

Furthermore, it emerged during discussions with Peter Ladkin, who is a pilot and domain expert, that some of the sentences whose syntax are described by the official manuals will never be used in an ATC dialogue.

3.5 Making the grammar complete and self-contained

The original Cushing Grammar is incomplete in diverse respects. In several rules, non-terminal symbols are used that are neither defined nor annotated as to which values they can have or which semantic category they represent. Although some of them are self-explanatory because of their names, they must be defined in a complete and machine-processable grammar.

Example

One simple example is the phrase definition

SQUAWK <code> [AND IDENT].

in the Cushing Grammar where the token <code> was never defined before. To fix that special problem we just had to define what is meant by "code" in this context.

```
# note: used in "SQUAWK <code>"
<code>               (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                     (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                     (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                     (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
```

A code consists of four digits each from zero up to seven.

Another kind of incompleteness of the Cushing Grammar that must be fixed is that some token definitions are empty. Some times Cushing only describes which kind of values the token should stand for. In this case explicit values or at least some example values must be inserted where to much possibilities are.

Example

In the Cushing Grammar the token <lname> includes only the information that "lname" stands for "location name" but no values are stated.

<lname> (location name)
Note: currently includes only a few cities

To tone that problem down we inserted at least a few exampling values.

```
# Location name of an airport
# note: different exampling values stated
<lname>            BOSTON | BALTIMORE | CHICAGO | ATLANTA | LOS ANGELES |
                   DÜSSELDORF | HANOVER | BERLIN | DIAPERBLEACH | VIENNA |
                   LONDON | MANCHESTER | DUBLIN
```

3.6 Making the grammar as correct as possible

To make the grammar as correct as possible, we derived in several iterative worksteps umpteen example sentences by recursive substitution of the non-terminal symbols. These sentences were evaluated by a domain expert as to whether they are correct or where we must seek for the error or the problem. With the results of this analysis (for details see [Elle2001]) we changed the grammar rules and performed these worksteps again with the new rules to make sure whether they still describe incorrect sentences.

Nevertheless, on some points it is very difficult or impossible to make some rules correct. To fulfil the aim, preferably not to reject any correct sentences, it makes sense to ease the restrictions for some rules on some cases. In certain circumstances this can have the negative side effect that also some incorrect sentences can be derived. We had to take this conflict into account when we consider changing the specific rule.

Examples

A frequency is a floating point number with three positions before and one to three positions after the decimal point. The positions after the decimal point are eighths or twentieths, i. e. in steps of .125 or in steps of .05. Thus the rule for the token `<frequency>` may be:

```
# Frequency (from 000.0 to 999.95)
# see:  AIM 4-2-8e, 4-2-3a2, 4-2-3d2
# note: <digit> and <digit++> put in concrete terms
<frequency>          <digit> <digit> <digit> <point> (<digit> [<5>] | <eighths>)
```

But not every frequency that is possible concerning this rule can be used in a real ATC communication at any time of the dialogue. So actually the possible values for a frequency must be adapted for the specific context.

Similar problems are the definitions for the registration number of an aircraft and the short form of it:

```
# Registration number
<reg_number>         <country_design> <reg_design>

# Country designator
<country_design>     <alphanum>

# Registration designator
<reg_design>         <alphanum>

# Short registration number
# note: a suffix of the registration number consisting of two or three letters
#       or digits
<short_reg>          <alphanum>
```

The registration number which is used when giving the callsign to identify the aircraft consists of a country designator and the registration designator. In general the country and the registration designator are alpha-numerical sequences (represented by the token `<alphanum>`), i. e. sequences of digits and letters. But in reality the syntax of a registration designator is different from country to country; even the country designator does not have a uniform syntax.

The problem of the short registration number as often used in ATC calls is even more difficult. It is a suffix of the registration number. So if an ATC grammar should be perfect at this point it has to check whether a short registration number is consistent with the suffix of a registration number that is mentioned in the context of the dialogue. But to fulfil this purpose one needs a context-sensitive grammar, not only a context-free grammar such as the one we developed.

4 Reflections and Conclusion

The strategy that we have chosen is not the only way to develop a new and better ATC grammar fulfilling the requirements. But it shows several important aspects that should be taken into account when trying to design a grammar for a part of the ATC language or – more general – when trying to transform a vaguely formulated grammar into a machine processable form.

It is our experience that an iterative development of development and inspection/critique is needed. We needed four iterations to achieve our goal satisfactorily. This may be because the requirements and problems they engender are highly interrelated and influence or even exclude one another.

The grammar we constructed is formally complete (as defined in section 2) but is not yet completely correct. We resolved the conflict between correctness and completeness in favour of final completeness. It also describes only a restricted extract of all possible ATC/pilot communication. But because of its modularity, it is unquestionably easier to revise our grammar once more, to debug it or to extend or rewrite it for other purposes.

Bibliography

References

[Cush1994] Steven Cushing, *Fatal Words*,
University of Chicago Press, 1994

[Bell2000] Sharlene Bell, Jean Lanigan, John Groarke, *DCG Parser for ATC Language*,
Project Report, University of Bielefeld, 2000

[Elle2001] Martin Ellermann, Mirco Hilbert, *Review of the Cushing Grammar*,
Technical Report RVS–Occ–01–02, RVS Group, Faculty of Technology,
University of Bielefeld, 2001

[Schö1995] Uwe Schöning, *Theoretische Informatik – kurzgefaßt*,
Spektrum Akademischer Verlag, Heidelberg/Berlin/Oxford, 1995

[AIM] Aeronautical Information Manual
Official Guide to Basic Flight Information and ATC Procedures
`http://www.faa.gov/ATPubs/AIM/`

[ATC] Air Traffic Control
`http://www.faa.gov/ATPubs/ATC/`

Resources

[FAA] American Federal Aviation Administration Academy
`http://www.atctraining.faa.gov/site/`

[Airdis] Airdisaster.com
`http://www.airdisaster.com`

[ASN] Aviation Safety Network
`http://aviation-safety.net`

A ATC Grammar by Martin Ellermann and Mirco Hilbert

A.1 Syntax for Grammar Definition

The syntax used in the grammar specification given below is as follows:

- All uppercase letters indicate that the words are to be spoken verbatim.
- All lowercase letters, numbers and underscores enclosed in angle brackets (<...>) indicate non-terminals, that is, variables whose syntax has been previously defined.
- Brackets ([...]) indicate that the enclosed data may or may not be applicable.
- Parentheses ((...)) are used for grouping.
- A vertical line (|), generally used in conjunction with parentheses, indicates that one of the two groups of words the vertical line separates is to be selected.
- A plus sign (+) indicates one or more repetitions of the preceding token.
- A hash (# ...) indicates a comment on the following grammar rule, a description of what is to be said or a hint to an official manual where to find more information about the rule.

A.2 Air Traffic Control Grammar (version 0.9)

A.2.1 Token Definitions

```
# Several information about the different token definitions are from the
# "Aeronautic Information Manual" (AIM), chapter 4 "Air Traffic Control",
# section 2 "Radio Communications Phraseology and Techniques"
# or from the "Air Traffic Control" handbook (ATC)
# of the "American Federal Aviation Administration Academy" (FAA)
# ( http://www.faa.gov/ATPubs/AIM/ resp. http://www.faa.gov/ATPubs/ATC/ )
# [
#   - non-terminals were normalized so that they are only allowed to consist
#     of non-capital letters, digits and underscores
#     ( as a regular expression:  \<[a-z0-9_]+\> )
#   - "<digit+1>" and similars were consequently replaced by "<digit> <digit>"
#   - "<digit++>" and similars were consequently replaced by "<digit_seq>",
#     that should stand for a sequence of digits
# ]

<0>                   0 | ZERO
<1>                   1 | ONE
```

```
<2>                 2 | TWO
<3>                 3 | THREE
<4>                 4 | FOUR
<5>                 5 | FIVE
<6>                 6 | SIX
<7>                 7 | SEVEN
<8>                 8 | EIGHT
<9>                 9 | NINE | NINER | NINA
<10>                1 0 | TEN
<11>                1 1 | ELEVEN
<12>                1 2 | TWELVE

<point>             . | POINT


# Positiv digits (without 0)
<pos_digit>         <1> | <2> | <3> | <4> | <5> | <6> | <7> | <8> | <9>


# All digits (with 0)
<digit>             <0> | <pos_digit>

<digit_seq>         <digit> [<digit_seq>]


# Numbers without leading zeros
<number>            <0> | <pos_digit> [<digit_seq>]


# Numbers without leading zeros greater than one
<number_gt_one>     <2> | <3> | <4> | <5> | <6> | <7> | <8> | <9> |
                    <pos_digit> <digit_seq>


# Letters
<letter>            A | B | C | D | E | F | G | H | I | J | K | L | M |
                    N | O | P | Q | R | S | T | U | V | W | X | Y | Z

<letter_seq>        <letter> [<letter_seq>]


# Spoken letters
<spoken_letter>     ALFA | BRAVO | CHARLIE | DELTA | ECHO | FOXTROT | GOLF |
```

```
                     HOTEL | INDIA | JULIETT | KILO | LIMA | MIKE |
                     NOVEMBER | OSCAR | PAPA | QUEBEC | ROMEO | SIERRA | TANGO |
                     UNIFORM | VICTOR | WHISKEY | XRAY | YANKEE | ZULU

<spoken_letter_seq> <spoken_letter> [<spoken_letter_seq>]


# Alpha-numerical sequence
<alphanum>           (<digit> | <letter>)+


# Specification of angles in degrees (from 001 to 360)
<degrees>            <0> <0> <pos_digit> |
                     <0> <pos_digit> <digit> |
                     (<1> | <2>) <digit> <digit> |
                     <3> (<0> | <1> | <2> | <3> | <4> | <5>) <digit> |
                     <3> <6> <0>


# Eighths
<eighths>            <1> <2> <5> | <2> <5> | <3> <7> <5> | <5> |
                     <6> <2> <5> | <7> <5> | <8> <7> <5>


# Facility function
# note: different exampling values stated
<ffunction>          APPROACH | DEPARTURE | CENTER | TOWER


# Location name of an airport
# note: different exampling values stated
<lname>              BOSTON | BALTIMORE | CHICAGO | ATLANTA | LOS ANGELES |
                     DÜSSELDORF | HANOVER | BERLIN | DIAPERBLEACH | VIENNA |
                     LONDON | MANCHESTER | DUBLIN


# Specification of clock (from 1 to 12)
<clock_az>           (<pos_digit> | <10> | <11> | <12>) O'CLOCK


# Cardinal points
<direction>          <quad> | <location>
```

```
<quad>              NORTH | SOUTH | EAST | WEST

<location>          NORTHEAST | NORTHWEST | SOUTHEAST | SOUTHWEST


# Miles specification
# note: formerly defined as <digit++> MILES
<miles>             <1> MILE | <number_gt_one> MILES


# Relative movement of other traffic
<rel_movement>      <rel_movement_pre> | <rel_movement_sub>

<rel_movement_pre>  CLOSING | CONVERGING | PARALLEL | OPPOSITE DIRECTION |
                    DIVERGING | OVERTAKING

<rel_movement_sub>  CROSSING (LEFT TO RIGHT | RIGHT TO LEFT)


# Bird species
<bird_species>      DUCKS  | GEESE  | GULLS | SPARROWS | CANARIES


# Bird size
<bird_size>         SMALL | LARGE


# Time specification
# see:  AIM 4-2-12
<time>              <hour> <minutes> ZULU

<hour>              (<0> | <1>) <digit> | <2> (<0> | <1> | <2> | <3>)

<minutes>           (<0> | <1> | <2> | <3> | <4> | <5>) <digit>


# Specification of altimeter (in inches Hg.)
# see:  ATC 2-7
# note: ALTIMETER THREE ONE ZERO ZERO means 31.00 inches Hg.
<altimeter>         ALTIMETER (<2> | <3>) <digit> <digit> <digit>


# Direction of the aircraft (in degrees)
```

```
# note: formerly defined as: HEADING <digit++>
<heading>            HEADING <degrees>


# Frequency (from 000.0 to 999.95)
# see:  AIM 4-2-8e, 4-2-3a2, 4-2-3d2
# note: <digit> and <digit++> put in concrete terms
<frequency>          <digit> <digit> <digit> <point> (<digit> [<5>] | <eighths>)


# Speed specification (in knots)
# see:  AIM 4-2-11
# note: >>controllers may omit the word "KNOTS" when using speed adjustment
#       procedures, e.g. "REDUCE/INCREASE SPEED TO TWO FIVE ZERO."<< neglected
<speed>              <1> KNOT | <number_gt_one> KNOTS


# Speed specification (in Mach)
# see:  AIM 4-2-11
<mach_number>        MACH [<1>] <point> <digit> [<digit>]


# Altitude
# note: <altitude> definition devided into low and high altitude
<altitude>           <altitude_low> | <altitude_high>

# Altitude (if the number of feet is less than 18,000 =>  from 0 to 17,900)
# see:  AIM 4-2-9a
# note: 1,000 feet = 304.80 m
#       <digit> and <digit++> put in concrete terms
<altitude_low>       ALTITUDE (
                        <digit> |
                        <1> (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                     ) THOUSAND [<digit> HUNDRED]

# Altitude (else =>  from 180 to 450)
# see:  AIM 4-2-9b
# note: FL 180 = 18,000 feet = 5,486.40 m
#       <digit++> put in concrete terms
<altitude_high>      FLIGHT LEVEL (
                        <1> (<8> | <9>) <digit> |
                        (<2> | <3>) <digit> <digit> |
                        <4> (<1> | <2> | <3> | <4>) <digit> |
```

```
                        <4> <5> <0>
                    )


# Facility name
# note: different exampling values stated
<fname>             LOGAN | JFK


# Type of an aircraft
# note: formerly named as: <craft-type> and defined as: DC-8/APACHE,
#       different exampling values stated,
#       A stands for Airbus and B for Boeing
<type_designator>   A 310 | A 320 | A 340 | B 747 | B 757 | B 777 |
                    DC-8 | MD-11


# Informal designation/nickname of an aircraft
# see:  AIM 4-2-4
# note: Prefix TANGO added for air taxi or other commercial operators not having
#       FAA authorized call signs
<callsign>          [TANGO] <airline> (<flight_number> | reg_number>)

# Short designation of an aircraft
<short_callsign>    <craft_type> <short_reg>

# Aircraft type, model or manufacturer's name
# note: different exampling values stated
<craft_type>        (
                        AIRBUS | APACHE | AZTEC |
                        BARON | BOEING | BONANZA | BREEZY |
                        CESSNA | CHEROKEE |
                        LEAR | LEARJET | LOCKHEAD |
                        MALIBU | MD |
                        SENECA
                    ) [HEAVY] |
                    (
                        APACHE | AZTEC |
                        BARON | BONANZA | BREEZY |
                        CESSNA | CHEROKEE |
                        DC |
                        LEAR | LEARJET |
                        MALIBU | MD |
```

```
                           SENECA | SITATION
                        ) [TENCE]

# Airline
# note: different exampling values stated, more airline names from
#       http://www.airlines.de/airlines/frame.html
<airline>               AERO ASIA | AERO LLOYD | AIR ATLANTIC | AIR BERLIN |
                        AIR CANADA | AIR CHINA | AIR EUROPA | AIR FRANCE |
                        ALASKA AIRLINES |
                        AMERICA WEST AIRLINES | AMERICAN AIRLINES |
                        BRITISH AIRWAYS |
                        CANADIAN AIRWAYS | CHINA AIRLINES | CONDOR |
                        DELTA AIR LINES |
                        JAPAN AIRLINES | JAPAN ASIA AIRWAYS |
                        KIWI INTERNATIONAL AIR LINES | KLM UK |
                        LUFTHANSA | LYNX AIR INTERNATIONAL |
                        OLYMPIC AIRWAYS | PACIFIC SOUTHWEST AIRLINES |
                        UNITED AIRLINES | US AIRWAYS

# Flight number
<flight_number>         <number>

# Registration number
<reg_number>            <country_design> <reg_design>

# Country designator
<country_design>        <alphanum>

# Registration designator
<reg_design>            <alphanum>

# Short registration number
# note: a suffix of the registration number consisting of two or three letters
#       or digits
<short_reg>             <alphanum>


# Flight route
# see:  ATC 4-4
<route>                 <route1> | <route2> | <route3> | <route4>

# ...
<route1>                VICTOR <number> [ROMEO | <location>]
```

```
# see:  AIM 4-2-8c
# note: J stands for jet route
<route2>            J <number> [ROMEO], <lmf_color> <number>

# ...
<route3>            NORTH AMERICAN ROUTE <number>

# ...
<route4>            (IR | VR) <number>

# Color of L/MF airway
<lmf_color>         RED | BLUE


# Navigational aid
# note: formerly defined as: VOR/VOR-TAC/TACAN/RADIO BEACON
#       VOR:     Very High Frequency Omni-directional Range
#       VOR-TAC: VOR / Tactical Air Navigation
#       TACAN:   Tactical Air Navigation
<navaid>            <navaid1> | <navaid2>

# Navigational aid that can be used in a position or fix specification
<navaid1>           WAYPOINT | V-O-R | BEACON |
                    [AUTO | MIDDLE | INNER] MARKER

<navaid2>           I-L-S | LOCALIZER


# ...
<fix>               <lname> (
                       <navaid1> | <degrees> RADIAL | <fix_azimuth>
                    ) |
                    <waypoint> | <dme_fix>

# ...
<dme_fix>           <miles> D-M-E FROM (V-O-R | I-L-S)

# ...
<waypoint>          LOCKO | HANKS | PINCH | RAYON | TIMMY

# note: A magnetic bearing extending from an Microwave Landing System (MLS)
#       navigation facility
```

```
<fix_azimuth>       <degrees> <direction> OF <lname> |
                    <degrees> RADIAL OF V-O-R |
                    <degrees> BEARING OF BEACON


# Weather level
<weather_level>     LEVEL <1> WEAK | LEVEL <2> MODERATE | LEVEL <3> INTENSE |
                    LEVEL <4> | LEVEL <5> | LEVEL <6> EXTREME


# Number of the runway (from 1 to 36)
<runway_num>        <pos_digit> | <10> | <11> | <12> | (<1> | <2>) <digit> |
                    <3> (<0> | <1> | <2> | <3> | <4> | <5> | <6>)


# note: used in "SQUAWK <code>"
<code>              (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                    (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                    (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)
                    (<0> | <1> | <2> | <3> | <4> | <5> | <6> | <7>)


# ...
<block_altitude>    BLOCK <altitude_low> THROUGH <altitude_low>


# note: used in "RADAR CONTACT [<position>]",
#       in some phrases Cushing used <fix> instead of <position>
<position>          [<miles> <direction> OF] (<fix> | <lname>)


# ...
<airway>            <route>
```

A.2.2 Block and Sentence Definitions

```
# At several block rules the flightphase in which this block could be said
# is specified. Flightphases are:
#    taxiing, takeoff, climb, cruise, approach, descend, landing, goaround,
#    emergency and urgency
# Also one or more categories are specified to that the block belongs. The
# category names correspond to the names defined by Andre Döring, Mark
# McGovern and Jan Sanders. They are:
#    heading, altitude, speed, position, radio operations, runway, markers,
#    weather, other planes and procedures

<atc_block>    <block01> | <block02> | <block03> | <block04> | <block05> |
               <block06> | <block07> | <block08> | <block09> | <block10> |
               <block11> | <block12> | <block13> | <block14> | <block15> |
               <block16> | <block17> | <block18> | <block19> | <block20> |
               <block21> | <block22> | <block23> | <block24> | <block25> |
               <block26> | <block27> | <block28> | <block29> | <block30> |
               <block31> | <block32> | <block33> | <block34> | <block35> |
               <block36> | <block37> | <block38> | <block39> | <block40> |
               <block41> | <block42> | <block43> | <block44> | <block45> |
               <block46> | <block47> | <block48> | <block49> | <block50> |
               <block51> | <block52>


# (Non-)affirmative answer to a question
# flightphase: all
# note: This phraseology was omitted from interface, because it requires a
#       pilot-initiated dialogue, which has not yet been addressed. Also,
#       AFFIRMATIVE/NEGATIVE could apply to any controller response to
#       pilot-initiated queries. (Cushing)
<block01>      <block01a> | <block01b>

<block01a>     <sentence01>

<sentence01>   AFFIRMATIVE.

<block01b>     <sentence02>

<sentence02>   NEGATIVE.


# Contact instruction
```

```
# flightphase: all
# category:     radio operations
<block02>       <sentence03>

<sentence03>    CONTACT (<fname> | <lname>) <ffunction> [<frequency>]
                [AT (<time> | <fix> | <altitude>)].


# Frequency setting
# flightphase: all
# category:     radio operations
<block03>       <sentence04>

<sentence04>    CHANGE TO MY FREQUENCY <frequency>.


# Frequency setting
# flightphase: all
# category:     radio operations
<block04>       <sentence05>

<sentence05>    REMAIN THIS FREQUENCY.


# Traffic
# flightphase: cruise
# category:     other planes
<block05>       <sentence06>

<sentence06>    TRAFFIC, (
                   <clock_az> |
                   <direction>, <miles>, [<quad> BOUND,] <rel_movement>,
                   [<craft_type>,] (
                      <altitude> |
                      ALTITUDE UNKNOWN
                   )
                ).


# Traffic
# flightphase: cruise
# category:     other planes
<block06>       <block06a> | <block06b>
```

```
<block06a>      <sentence07>

<sentence07>    TRAFFIC, (<miles> | <minutes> MINUTES) <direction> OF
                (<fname> | <fix>), <direction> BOUND, [<craft_type>,]
                (<altitude> | ALTITUDE UNKNOWN).

<block06b>      <sentence08>

<sentence08>    TRAFFIC, NUMEROUS TARGETS VICINITY (<fname> | <fix>).


# Traffic
# flightphase: cruise
# category:     other planes
<block07>       <sentence09> <sentence10>

<sentence09>    TRAFFIC ALERT [
                   <clock_az> |
                   <direction>, <miles>, [<quad> BOUND,] <rel_movement>
                ].

<sentence10>    ADVISE YOU [
                   TURN (LEFT | RIGHT) [<heading>] AND
                ]
                (CLIMB | DESCEND) [TO <altitude>] IMMEDIATELY.


# Traffic
# flightphase: cruise
# category:     other planes
<block08>       <sentence11>

<sentence11>    [<clock_az>] TRAFFIC NO LONGER A FACTOR.


# Birds
# flightphase: cruise
# category:
<block09>       <block09a> | <block09b>

<block09a>      <sentence12>
```

```
<sentence12>   FLOCK OF (<bird_species> | [<bird_size>] BIRDS), (
                  <direction> BOUND ALONG <route> |
                  <clock_az> <miles> <direction> BOUND |
                  VICINITY (<fname> | <fix>)
               ),
               (LAST REPORTED AT <altitude> | ALTITUDE UNKNOWN).

<block09b>     <sentence13>

<sentence13>   NUMEROUS FLOCKS (<bird_species> | [<bird_size>] BIRDS),
               VICINITY (<fname> | <fix>),
               (LAST REPORTED AT <altitude> | ALTITUDE UNKNOWN).


# Weather
# flightphase: all
# category:    weather
<block10>      <sentence14>

<sentence14>   REQUEST FLIGHT CONDITIONS (
                  IN REGION | OVER <fix> | ALONG PRESENT ROUTE |
                  BETWEEN <fix> AND <fix>
               ).


# Weather
# flightphase: cruise
# category:    weather
<block11>      <block11a> | <block11b> | <block11c> | <block11d> | <block11e>

<block11a>     <sentence15>

<sentence15>   WEATHER AREA BETWEEN <clock_az> AND <clock_az> <miles>.

<block11b>     <sentence16>

<sentence16>   <number> MILE BAND OF WEATHER FROM [
                  <miles> <direction> OF
               ] <fix> TO [
                  <miles> <direction> OF
               ] <fix>.

<block11c>     <sentence17>
```

```
<sentence17>    <weather_level> WEATHER ECHO BETWEEN <clock_az> AND
                <clock_az> <miles>, MOVING <direction> AT <number>
                KNOTS TOPS <altitude>.

# note: Deviation fragments are made more clear here. Official
#       manual is ambiguous on this point. (Cushing)
<block11d>      <sentence18> <sentence19>

<sentence18>    DEVIATION APPROVED.

<sentence19>    ADVISE WHEN ABLE TO (RETURN TO COURSE | RESUME NORMAL NAVIGATION).

# note: Deviation fragments are made more clear here. Official
#       manual is ambiguous on this point. (Cushing)
<block11e>      <sentence20> <sentence21>

<sentence20>    UNABLE DEVIATION.

<sentence21>    (FLY <heading> | PROCEED DIRECT TO <fix>).


# Weather
# flightphase: cruise
# category:    weather
<block12>       <sentence22> <sentence23>

<sentence22>    WIND SHEAR (
                   REPORTED | ALERT | ALERTS (TWO | SEVERAL | ALL) QUADRANTS
                ).

<sentence23>    CENTERFIELD WIND <direction> AT <speed> (
                   , <direction> BOUNDARY WIND <direction> AT <speed> |
                   VARYING TO <direction> AT <speed>
                ).


# ...
# flightphase: cruise
# category:    position
# note: Terminal control messages are not part of current interface. (Cushing)
<block13>       <sentence24>
```

```
<sentence24>   HOLD (SHORT OF RUNAWAY | IN POSITION).


# Clearance
# flightphase: cruise
# category:
# note: This message was used to group together all messages
#       starting with VIA or involving route assignment. (Cushing)
<block14>      <sentence25>

<sentence25>   CLEARED TO <fix> VIA ((<route> | <fix>)+).


# Clearance, direction setting
# flightphase: cruise
# category:
<block15>      <block15a> | <block15b>

<block15a>     <sentence26>

<sentence26>   CLEARED TO FLY <direction> OF <lname> <navaid> BETWEEN
               THE <number> AND THE <number> (
                  BEARINGS FROM | RADIALS
               ) WITHIN <number> MILE RADIUS.

<block15b>     <sentence27>

<sentence27>   CLEARED TO FLY <quad> QUADRANT OF <lname> <navaid>
               WITHIN <number> MILE RADIUS.


# Clearance, direction setting
# flightphase: cruise
# category:
<block16>      <sentence28>

<sentence28>   CLEARED TO FLY <direction> OF THE <lname> RUNWAY
               <runway_num> BETWEEN THE <number> AND THE
               <number> AZIMUTHS WITHIN <number> MILE RADIUS.


# Clearance
# flightphase: cruise
```

```
# category:
# note: CLEARED was added, and the fragment OFFSET was added to
#       the fragment DIRECT. The DIRECT fragment was generalized to
#       contain all valid fixes. (Cushing)
<block17>      <sentence29>

<sentence29>   CLEARED DIRECT TO THE <fix> [
                  , VIA <route> OFFSET <miles> (RIGHT | LEFT)
               ].


# Altitude setting
# flightphase: cruise
# category:    altitude
<block18>      <sentence30>

<sentence30>   MAINTAIN [CRUISE] ALTITUDE [
                  UNTIL (
                     <time> | PAST <fix> | <miles> PAST <fix> |
                     <minutes> MINUTES PAST <fix>
                  )
               ].


# Altitude setting
# flightphase: cruise
# category:    altitude
<block19>      <block19a> | <block19b>

<block19a>     <sentence31>

<sentence31>   [(CLIMB | DESCEND) AND] MAINTAIN <altitude> [
                  AFTER PASSING <fix> | AT <time> |
                  WHEN ESTABLISHED AT LEAST (<miles> | <minutes> MINUTES) PAST <fix>
               ].

<block19b>     <sentence32>

# note: AT <TIME> | <FIX> made optional in CLIMB TO REACH message. (Cushing)
<sentence32>   (CLIMB | DESCEND) TO <altitude> [AT (<time> | <fix>)].


# Altitude setting
```

```
# flightphase: cruise
# category:
# note: The phrase below allows all possible combinations of
#       <altitude> requirements in a CROSS <FIX> phrase. (Cushing)
<block20>      <sentence33>

<sentence33>   CROSS <fix> AT (
                  OR (ABOVE | BELOW) <altitude> | AND MAINTAIN <altitude>
               ).


# Altitude setting
# flightphase: cruise
# category:
# note: AT PILOT'S DISCRETION was implemented as an option. (Cushing)
<block21>      <sentence34>

<sentence34>   (CLIMB | DESCEND) AT PILOT'S DISCRETION.


# Clearance, altitude
# flightphase: cruise
# category:
<block22>      <sentence35>

<sentence35>   EXPECT (FURTHER | [FURTHER] (CLIMB | DESCENT)) CLEARANCE (
                  IN <miles> | IN <minutes> MINUTES | AT <fix>
               ).


# Altitude request
# flightphase: cruise
# category:    altitude
<block23>      <sentence36>

<sentence36>   REQUEST (
                  <altitude> [FROM] <fname> [AT (<time> | <fix>)] |
                  HIGHER | LOWER
               ).


# Clearance
# flightphase: cruise
```

```
# category:
# note: (routing) was assumed to refer to any combination of
#       routes and fixes used to describe a clearance. (Cushing)
<block24>       <sentence37>

<sentence37>    EXPECT FURTHER CLEARANCE VIA ((<fix> | <route>)+).


# Clearance
# flightphase: cruise
# category:
<block25>       <block25a> | <block25b> |  <block25c> |  <block25d>

<block25a>      <sentence38>

# note: both <fix>-specifications must have the same values
<sentence38>    CLEARED TO <fix>, HOLD AT <fix> <direction>, AS PUBLISHED.

<block25b>      <sentence39>

<sentence39>    CLEARED TO <fix>, NO DELAY EXPECTED.

<block25c>      <sentence40>

<sentence40>    CLEARED TO <fix> VIA LAST ROUTING CLEARED.

<block25d>      <sentence41>

# note: the first two <fix>-specifications must have the same values
<sentence41>    CLEARED TO <fix>, HOLD AT <fix> <direction> OF <fix> ON THE (
                   <degrees> RADIAL | <degrees> COURSE | <degrees> BEARING |
                   <degrees> AZIMUTH | <route>
                )
                [, <number> MILE LEGS] [, (LEFT | RIGHT) TURNS].


# Clearance
# flightphase: cruise
# category:
# note: (time) was changed to AT <TIME>|<fix>. (Cushing)
<block26>       <sentence42>

# note: en route = unterwegs
```

```
<sentence42>   EXPECT FURTHER CLEARANCE AT (<time> | <fix>) [
                  ANTICIPATE ADDITIONAL (<minutes> MINUTE | <hour> HOUR)
                  (DELAY AT <fix> | EN ROUTE DELAY | TERMINAL DELAY)
               ].


# Clearance
# flightphase: cruise
# category:
# note: DELAY INDEFINITE was made an optional preface to all EXPECT messages.
#        (Cushing)
<block27>      <sentence43>

<sentence43>   DELAY INDEFINITE, EXPECT FURTHER CLEARANCE <time>.


# Position
# flightphase: cruise
# category:    position
# note: OVER/PASSING <fix> was changed to PASSING <fix> to avoid confusion
#        with OVER that is used in parser to signify end of message (transmit).
#        Also, note that | SHOW YOUR POSITION AS was added as a preface to make
#        these into controller messages, where usually they are pilot responses
#        to controller queries. (Cushing)
<block28>      <sentence44>

<sentence44>   SHOW YOUR POSITION AS (
                  PASSING <fix> | <miles> FROM <fix> |
                  <miles> <direction> OF (<fix> | <lname>) |
                  (CROSSING | JOINING | DEPARTING) <route> AT <miles> FROM <fix> |
                  (INTERCEPTING | CROSSING) <lname> <degrees> RADIAL
               ).


# Radar
# flightphase: cruise
# category:    radio operations
<block29>      <sentence45>

<sentence45>   SQUAWK (<code> [AND IDENT] | STANDBY).


# Radar
```

```
# flightphase: cruise
# category:    radio operations
<block30>      <sentence46>

<sentence46>   STOP SQUAWK.


# Radar
# flightphase: cruise
# category:    radio operations
<block31>      <sentence47>

<sentence47>   RADAR CONTACT [<position>].


# Radar
# flightphase: cruise, emergency
# category:    radio operations
<block32>      <sentence48>

<sentence48>   SQUAWK 7700.


# Radar
# flightphase: cruise
# category:    radio operations
<block33>      <sentence49>

<sentence49>   RADAR SERVICE TERMINATED. [SQUAWK <code>.]


# Radar
# flightphase: cruise
# category:    radio operations
<block34>      <sentence48> <sentence49>

<sentence48>   PRIMARY RADAR OUT OF SERVICE.

<sentence49>   TRAFFIC ADVISORIES AVAILABLE ON TRANSPONDER-EQUIPPED AIRCRAFT ONLY.


# Radar
# flightphase: cruise
```

```
# category:     radio operations
<block35>       <sentence50>

<sentence50>    RESET TRANSPONDER.


# Radar
# flightphase: cruise
# category:     radio operations
<block36>       <sentence51>

# see:  ATC 5-2-14
<sentence51>    YOUR TRANSPONDER APPEARS (INOPERATIVE | MALFUNCTIONING),
                RESET, SQUAWK <code>.


# Radar
# flightphase: cruise
# category:     radio operations
# note: syntax for (alternative instructions when required) was not known,
#       so it was omitted in the grammar fragment above. (Cushing)
<block37>       <sentence52>

<sentence52>    RADAR CONTACT LOST.


# Altitude setting
# flightphase: cruise
# category:     altitude
<block38>       <sentence53> <sentence54> <sentence55>

<sentence53>    LOW ALTITUDE ALERT.

<sentence54>    CHECK YOUR ALTITUDE IMMEDIATELY.

<sentence55>    THE (
                   ((M-E-A | M-O-C-A | M-I-A) IN YOUR VICINITY) |
                   M-D-A
                )
                IS <altitude>.


# Altitude
```

```
# flightphase: cruise
# category:    altitude
<block39>      <sentence56>

<sentence56>   SAY ALTITUDE.


# Altitude
# flightphase: cruise
# category:    altitude
<block40>      <sentence57>

<sentence57>   VERIFY ALTITUDE AND ALTIMETER SETTING.


# Altitude, radar
# flightphase: cruise
# category:    altitude
<block41>      <block41a> | <block41b>

<block41a>     <sentence58>

<sentence58>   SQUAWK ALTITUDE.

<block41b>     <sentence59>

<sentence59>   STOP ALTITUDE SQUAWK.


# Altitude, radar
# flightphase: cruise
# category:    altitude
<block42>      <sentence59> <sentence60>

<sentence60>   ALTITUDE DIFFERS BY <number> FEET.


# Altitude
# flightphase: cruise
# category:    altitude
<block43>      <block43a> | <block43b>

<block43a>     <sentence61>
```

```
<sentence61>   VERIFY AT <altitude>.

<block43b>     <sentence62>

<sentence62>   VERIFY ASSIGNED ALTITUDE <altitude>.


# Direction setting
# flightphase: cruise
# category:    heading
# note: FLY <heading> and FLY PRESENT HEADING were implemented as a choice
#       at the end of a VECTOR phraseology. (Cushing)
<block44>      <block44a> | <block44b> | <block44c> | <block44d>

<block44a>     <sentence63>

<sentence63>   TURN (LEFT | RIGHT) <heading>.

<block44b>     <sentence64>

<sentence64>   FLY <heading>.

<block44c>     <sentence65>

<sentence65>   FLY PRESENT HEADING.

<block44d>     <sentence66>

<sentence66>   DEPART <fix> <heading>.


# Direction setting
# flightphase: cruise
# category:    heading
<block45>      <sentence67>

<sentence67>   TURN <degrees> DEGREES (LEFT | RIGHT).


# Direction setting
# flightphase: cruise
# category:    heading
```

```
<block46>       <block46a> | <block46b>

<block46a>      <sentence68>

<sentence68>    THIS WILL BE A NO-GYRO VECTOR, (
                   TURN (LEFT | RIGHT) |
                   START (LEFT | RIGHT) TURN
                ).

<block46b>      <sentence69>

<sentence69>    STOP TURN.


# Direction setting
# flightphase: cruise
# category:     heading
<block47>       <block47a> | <block47b> | <block47c> | <block47d>

<block47a>      <sentence70>

<sentence70>    VECTOR TO (<fix> | <airway>), FLY <heading>.

<block47b>      <sentence71>

<sentence71>    VECTOR TO INTERCEPT <lname> <degrees> RADIAL, FLY <heading>.

<block47c>      <sentence72>

<sentence72>    VECTOR FOR SPACING, FLY <heading>.

<block47d>      <sentence73>

<sentence73>    VECTOR TO <lname> FINAL APPROACH COURSE, FLY <heading>.


# Direction setting
# flightphase: cruise
# category:
# note: (position with respect to course|fix) removed from
#       RESUME OWN NAVIGATION. Also, WHEN ABLE ... removed from
#       FLY <HEADING> - it is implemented as part of DEVIATION messages. (Cushing)
<block48>       <sentence74> <sentence75>
```

```
<sentence74>    RESUME OWN NAVIGATION.

<sentence75>    WHEN ABLE, PROCEED DIRECT <fix>.


# Speed setting
# flightphase: cruise
# category:     speed
# note: Confusion among homonyms eliminated by revision.
#        IF PRACTICAL, was added as an optional preface. (Cushing)
<block49>       <block49a> | <block49b> | <block49c> | <block49d> | <block49e>

<block49a>      <sentence76>

<sentence76>    (ACCELERATE | [IF PRACTICAL,] SLOW) TO
                (SPEED <speed> | <mach_number>).

<block49b>      <sentence77>

<sentence77>    (INCREASE | [IF PRACTICAL,] REDUCE) SPEED (BY | TO)
                (<number> KNOTS | <mach_number>).

<block49c>      <sentence78>

# see:  ATC 5-7-2
<sentence78>    SAY (AIRSPEED | MACH NUMBER).

<block49d>      <sentence79>

<sentence79>    MAINTAIN PRESENT SPEED.

<block49e>      <sentence80>

<sentence80>    DO NOT EXCEED <speed>.


# Altitude setting, speed setting
# flightphase: cruise
# category:     speed, altitude
<block50>       <block50a> | <block50b>

<block50a>      <sentence81> <sentence82>
```

```
<sentence81>   SLOW TO (SPEED <speed> | <mach_number>). |
               REDUCE SPEED BY (<number> KNOTS | <mach_number>).

<sentence82>   DESCEND AND MAINTAIN <altitude>.

<block50b>     <sentence82> <sentence81>


# Altitude setting, speed setting
# flightphase: cruise
# category:    altitude, speed
<block51>      <sentence83>

<sentence83>   MAINTAIN (<altitude> | <block_altitude>) [AT <speed>].


# Speed setting
# flightphase: cruise
# category:    speed
<block52>      <sentence84>

<sentence84>   RESUME NORMAL SPEED.
```