ICAO Field 15 Parser

The ICAO Compliant field 15 parser parses ICAO field 15; the parser performs the following;

- Analyzes field 15 for correct syntax;
- Analyzes field 15 for correct semantics;
- Extracts a route from field 15;
- Reports accurate and meaningful errors should field 15 not conform to ICAO standards;

The software provides a solution for a relatively complex problem relieving your organisation from having to spend time and money in re-inventing the wheel.

Design Paradigm

The field 15 parser has been implemented using standard industry practice compiler technology. The Backus Naur Form (BNF) has been derived from the ICAO documentation and a suitable parser implemented using the BNF language definition.

Using standard industry compiler technology based on the BNF description of field 15 results in an extremely efficient parser with a small source code foot print; this ensures the implementation is fast, reliable and easily configurable.

It was always the objective of our company to make the software hardware and OS independent which is why we chose to write the parser in Java [tm]. Using Java [tm] also ensured that XML could be used for the output data which in turn ensures ease of integration with software utilizing the parser.

The parser is data driven and can be easily modified for custom implementations.

Parser Operation

The input to the parser is the field 15 ‘string’; the field must contain all sub-fields (i.e. ‘a’, ‘b’ and ‘c’). The output differs on the language in which the parser is implemented:

- ‘C’ – The output is an initialised structure that is passed to the parser along with the input string;
- Java – The output is provided as an XML document;

In both cases the output contains a list of ‘elements’ that start and end with the ADEP and ADES respectively and include all intermediate points; the points are connected with routes, SIDs, STARs, DCT or VFR ‘elements’. A rudimentary profile is derived using the speed and height given in field 15 (sub-fields ‘a’ and ‘b’). Speed and height changes defined along the route are also taken into account.

The parser derives and returns the flight rules based on the routing information; this can be used to consistency check a route description with the flight rules given in ICAO field 8a.

Software Details

The parser is implemented as a table driven finite state machine that is defined by the BNF. Using a table driven solution rather than a recursive descent parser makes the parser extremely easy to configure for custom implementations or upgrades to comply with the latest ICAO directives. Adding and/or modifying the behaviour of the parser based on changes in semantic or syntactical rules typically takes a few hours to complete.

The parser can be provided in Java [tm], ‘C’ or ‘C++’.

The ‘C’ implementation is approximately 1k lines of code and is supplied as a library file; the Java implementation is slightly larger and is provided as a single ‘jar’ file. Source code can also be supplied if required at extra cost. The software is provided with comprehensive API documentation.

Licensing

The libraries are available for a one off licence cost, in addition, maintenance support can also be provided. The maintenance support is foreseen to assist with integration and customisation, the latter is not uncommon in the ATM domain.

The source code can be provided but the costs are considerably higher and based on the foreseen usage of the product.