Project Documentation

Useful feedback in the Ampersand parser

Phase 3d

Daniel S. C. Schiavini and Maarten Baertsoen

Open Universiteit Nederland, faculteit Informatica
T61327 - Afstudeerproject bachelor informatica

June 11, 2015

Version 0.1

Contents

List of Figures

List of Tables

1 Introduction

1.1 Identification

This document contains the documentation of the project 'Useful feedback in the Ampersand parser'. The document is the milestone product of the project phase 3d and contains the deliverables 'analysis & design' and 'test report', as described in the project planning [?].

This document is part of the graduation project of the computer science bachelor at the Open Universiteit Nederland. The project 'Useful feedback in the Ampersand parser' is executed in collaboration with Maarten Baertsoen, with support of the supervisor Dr. Bastiaan Heeren and examiner Prof.dr. Marko C.J.D. van Eekelen. The assignment is given by Prof.dr. Stef Joosten, who researches how to further automate the design of business processes and information systems by the development of the Ampersand project.

Ampersand is an approach for the use of business rules to define the business processes. Users describe the business rules in a formal language (ADL), and Ampersand compiles those rules into functional specification, documentation and working software prototypes. The main objective of this project is to improve the feedback and maintainability of the Ampersand parser. See [?] for more details on the project.

1.2 Goals

The goals of this document are to provide enough documentation to support the further development of the Ampersand parser after this project is concluded. In order to provide sufficient knowledge, the following information is given:

- Describe the analysis of the new Ampersand parser;
- Explain the design decisions taken during the analysis and development;
- Show how the solution fulfills the project requirements;
- Describe how the system was tested in a test report.

1.3 Document overview

This initial section gives an introduction to the document. In section 2, the analysis of the parser is described. Afterward, in section 3, the software design is depicted, and in section 4 the test report is found. A short conclusion is given in section 5

Finally, in the appendix, a glossary of terms, definitions and abbreviations is given, just as a list of references.

2 Analysis

In this section, we analyze the previous Ampersand parser in order to understand its workings and signal the improvement points.

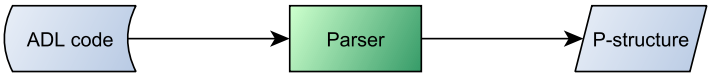


Figure 1: Relevant data flow for the Ampersand parsing component

2.1 System overview (R-M)

The requirements of the new parser have been described in the project planning [**?**]. Basically, both the old and the new parser must be able to make a conversion from a text file (ADL-script) to a parse tree (the P-structure). Figure 1 depicts this data flow.

Often, the parsing component is separated into a lexer (that converts text to tokens) and the actual parser (that converts the tokens into the parse tree). Since this separation is considered beneficial for both maintainability and performance [**?**], we assumed from the beginning that the new Ampersand parser would be separated in this way. This is depicted in Figure 2. The previous parser also had a separate lexer, but the name was scanner instead.

Figure 2: Data flow for the Ampersand lexing and parsing components

In order to take the next steps and understand how the parser can be designed, we first take a look at the grammar in subsection 2.2 and the parse tree in subsection 2.3. Afterwards, we analyze the lexer with the original token structure, so that we can define a new token structure, in subsection 2.4. Finally, we analyze the parser in subsection 2.5 and the generated errors in subsection 2.6.

2.2 Grammar

2.2.1 Getting the EBNF in good shape

The Ampersand grammar is described using the EBNF notation. EBNF is a notation technique with the goal to express a context free grammar like Ampersand. At the beginning of the project, we noticed that the existing EBNF diagram was outdated and not in line anymore with the actual syntax of Ampersand. As the EBNF is the crucial source of information in building the new parser, the first focus was to update the old EBNF to represent the actual Ampersand Syntax.

2.2.2 The actual EBNF diagram

Through reverse engineering, we checked all Haskell functions on the actual syntax they implement. In the source of the new parser, all the grammar expressions are placed above the actual parser function as code annotations to support code maintainability.

The derived syntax is up to date and visualized using a railroad diagram, an ideal technique to create a visual representation of context free grammars. Several railroad diagram generators are available on the internet, free of charge. We used the railroad diagram generator created by Gunter Rademacher, available on `http://bottlecaps.de/rr/ui`. The generated diagrams with the corresponding EBNF productions are available in the section 5.2.

One interesting plus is that during the project we found a bug in the Railroad Diagram Generator. The tool would crash with the `Trm4` expressions. This bug was reported to the author Gunther Rademacher, who promptly fixed the issue.

The EBNF diagram is available in appendix of this document. todo: reference

2.3 Parse tree (R-M)

The parse tree (also known as P-structure) is a data structure that very much resembles the EBNF description. The root of the tree is the `P_Context` structure, and every leaf of the tree has a field for the location where it was found in the ADL code (the `Origin` structure). The tree is consistently defined with the record syntax and is well documented.

However, the constructions are not completely pure, since some transformations are necessary from the ADL to the P-structure. This forces the parser to do more than only parsing. Also, the order of the fields can be confusing; sometimes `Origin` is the first field and sometimes it is not.

During this project, small changes to the parse tree have been done. These changes are described in subsection 3.4.

2.4 Lexer

The lexer module is responsible to split up the input stream into tokens. Tokens are meaningful pieces of the input string that can be recognized by the parser.

The following improvement points were identified after analysis:

Dispersed error messages The error messages produced by the lexer are of good quality. Each error message is however defined directly within the corresponding lexer function making the maintenance harder.

Complex token structure The token structure is complex and confusing. Two values are present in the token, of which one `val1` is never used. There is no distinction between the values used to identify the content of the token and the ones to determine the position of the token.

Module structuring In the lexer, the actual lexing functions are intermingled with data types, supporting functions and error message texts. This makes the lexer hard to understand and to maintain.

Language support The errors are returned in English only, no multilingual support is available.

No support for warnings The lexer can only return errors, warnings are not supported.

Strings only Token values are stored as strings for all types, with no conversion of integer values.

Lacking documentation There was no documentation available on how the lexer was designed and structured.

2.4.1 Token structure

The old token has the following structure:

```
data Token = Tok { tp' :: TokenType
                 , val1 :: String
                 , val2 :: String
                 , pos :: !Pos
                 , file :: !Filename
                 }

data TokenType
  = TkSymbol
  | TkVarid
  | TkConid
  | TkKeyword
  | TkOp
  | TkString
  | TkExpl
  | TkAtom
  | TkChar
  | TkInteger8
  | TkInteger10
  | TkInteger16
  | TkTextnm
  | TkTextln
  | TkSpace
  | TkError
  deriving (Eq, Ord)
```

The arguments have the following purpose:

TokenType Identification of the token type.

val1 This string argument is not used in the lexer. In the case of a `keyToken` creation, the value is filled in, but we could not find any purpose for this argument.

val2 The actual token content, stored as a string, including the integer values.

pos Line and column number.

file Filename in which the token is located.

2.5 Parser (R-M)

The previous Ampersand parser was generally well organized, so each ENBF rule could be mapped to a different parser. However, several flaws were observed as improvement points. During this project, we focused on the following issues:

Lacking documentation There was no documentation on the recognized grammar. The last EBNF available was not updated in a long while.

Ad-hoc transformations The parser was built with the applicative interface of the uulib. The applicative operators were thus used in sequence to recognize each of the accepted grammar productions. However, the parser was often forced to change the order and format of the parsed structures, because the parse tree did not match the grammar productions (i.e. many rebuild functions).

Long file Since all the grammar constructions – plus help functions – were in a single file, the parser was hardly readable. It summed a total of 823 lines of code only in the `Parser` module.

Pretty printing It used to be impossible to print the parse tree back to ADL-code. That made it harder to develop and test the parser properly.

Test suite There were no automated tests for the parser. Because of this, any code change would be hard to test and could potentially influence other Ampersand modules.

Duplicated code A large part of the code was duplicated and not used (mainly in the `Parsing` module).

Error messages As mentioned, the main reason for this project was the bad quality of the error messages generated. subsection 2.6 expands on this point.

Although it may be hard to resolve all the mentioned issues, we believe our efforts have played off well. In subsection 3.3 we describe how the new parser was designed.

2.6 Errors

In this section, we analyze the error messages given by the old parser. The results of this analysis are compared to the new parser in subsection 3.5.

2.6.1 Error message qualification

The user friendliness and correctness of an error message is a subjective topic and therefore we need to start with a definition to objectively judge the quality of an error message. After analyzing the current Ampersand parser error messages, we identified the following objective aspects of an Ampersand parser error message:

Position Each error messages is accompanied with the correct position (file, line number and column) of where the error was found.

Accuracy The accuracy of an error message is measured based upon the following characteristics:

1. **How does the provided error description exactly outlines the discovered error:** Providing the user with a good description of the encountered syntax issue will support a fast error resolution. When the issue is vaguely described without pinpointing the exact issue, the error resolution will be time consuming.
2. **Pinpointing the correct error**: An error can invoke multiple subsequent issues. These issues are however irrelevant for the user and the Ampersand parser should provide the exact origin of the issues.
3. **Quality of the hint:** The parser provides a hint for a solution together with the error message to support the user with the error resolution.

Conciseness Providing a good error description is one thing, but this one message can be hidden between several other error messages that result from the initial error. It is unlikely that users will easily find the exact originating issue in their source file when they are overwhelmed with a multitude of error messages.

Based on the objective Ampersand parser error characteristics, we defined the following definition to distinguish between good, bad and average error messages:

Bad error message A message is considered to be bad if one of the criteria blow is fulfilled:

Position The position has a deviation of more then 1 line or 10 column positions from where the actual error is made.

Accuracy

- The provided error description is useless for the user to determine the actual error.
- The provided error description is not appointing the main, originating error without any correlation towards this main error.

Conciseness More then distinct 3 errors are mentioned by the Ampersand parser.

Average error message A message is considered to be of average quality if one of the criteria blow is fulfilled:

Position The position has a deviation between 5 and 10 columns positions from where the actual error is made.

Accuracy

- The provided error description is not an exact description of the error but provides however useful information to discover the actual issue.
- The provided error description is not appointing the main, originating error but the link to the actual error can be discovered based on the provided information.
- The provided hint is incorrect.

Conciseness 2 or 3 errors are mentioned by the Ampersand parser.

Good error message We can state that any error message being not bad nor average is good.

2.6.2 Gathering process

To gather the necessary input for the as-is analysis, an exhaustive list of all possible error messages is created. This as-is analysis will be used as a reference base to verify the implementation of the new error mechanism with Parsec. The errors are invoked by simulating all possible syntax errors that will invoke an error within the Ampersand parser. Each syntax statement is therefore manipulated, introducing one specific error per time, and the resulting error message is then recorded together with the actual erroneous statement. The exact same statements are afterwards pushed through the new parser, making it possible to make a quantitative 'before and after' analysis. Special attention is given to avoid redundant errors that could influence the quantitative analysis. An example of such an redundant error is the use of a capital letter in defining a specific reference. Although these references are used in several syntax statements, there is only one procedure in the parser to check all references starting with a capital letter. An improvement in the error message of this check may only be taken into account one time.

2.6.3 Results

Based on our error message qualification definition, Table 1 visualizes the results of the as-is analysis. This analysis clearly confirms the statement that the quality of the error messages is of low quality and that there is a lot of room for improvement.

Error quality	Old parser	
Good	19	22,35%
Average	18	21,18%
Bad	48	56,47%
Total	**85**	**100,00%**

Table 1: Error message as-is analysis results

3 Design

3.1 System overview (D)

The parser module overview is given in Figure 3. Each of the modules are described in the following subsection. TODO: Add the exported functions to each module.

3.1.1 Modules

In this section a short description of each module is given:

Parsing module that implements the interface of the parser with the rest of the system. It is responsible for reading the input files, calling the lexer and the parser and returning a parse tree as result (or a parse error).

Parser module responsible for executing the parsing itself. It accepts the tokens that are allowed in each grammar production and generates the corresponding parse tree. The parser is described in subsection 3.3.

ParsingLib library that contains several useful functions to assist the parser, e.g. token recognition. These functions are not depending on the specific grammar rules.

ParseTree external module containing the parse tree data structures. Only details of this module have been changed during this project (e.g. field ordering).

PrettyPrinters contains the `Pretty` class and the functions responsible for printing the parse tree to ADL scripts in a 'pretty' way.

CtxError contains the data structures responsible for the parse errors and their location. This module has not been refactored as a part of this project.

Lexer module responsible for recognizing the input characters and converting them to tokens. The lexer, together with its sub-modules, is described in subsection 3.2.

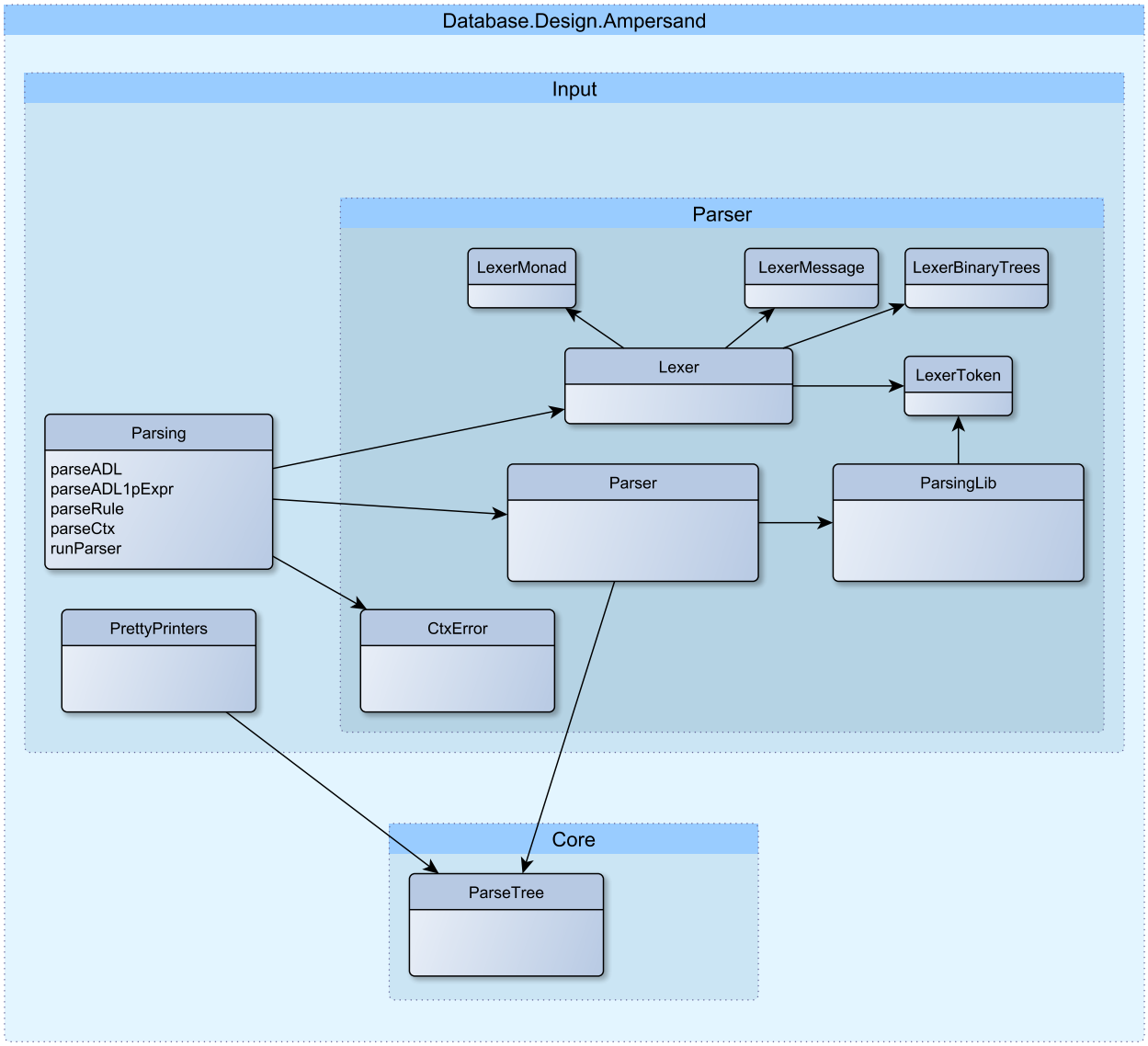


Figure 3: The parser modules and their relationships

3.2 Lexer

3.2.1 The rationale behind the new lexer

In the design of the new Ampersand parser, the question arose whether to keep the current scanner, the former name of the lexer module within Ampersand, or to implement a new one. After the analysis of the error improvement areas, as described in section 2, the main improvements were identified within the actual parser. The error feedback quality produced by the scanner module was higher and therefore, there was no stringent need to re-implement the scanner. On the other hand, given the aspect that Parsec was defined as the new parser library, keeping the current scanner would have resulted in the utilization of two different libraries providing more of less the same functionality. To avoid a decrease in maintainability, the decision is made to implement the parser and scanner based on the same library.

During the implementation of the lexer module, replacing the old scanner, additional attention was given to further improve the quality of the error messages. The scanner module is renamed to lexer to stress the aspect that the principle of lexemes is used in the new scanner.

Lexemes can be seen as the part of a token containing the actual language content besides the actual position information.

The lexer is built based on the existing Helium lexer modules. Helium is a Haskell compiler with the main goal of giving user friendly error messages [**?**]. The lexer module in Helium contains interesting principles such as position monitoring, warnings and easy maintainable error messages.

3.2.2 Lexer structure

The lexer is the main module, in which the actual lexing is done, and to do so, it uses the following sub-modules:

LexerMonad contains a monad definition that supports lexing with context. It tracks for example the location in the input and the warnings that may be generated. This module is re-used without any modification out of the Helium parser. The following functions or types are used in the Ampersand lexer:

- **LexerMonad** is the main monadic type used in the lexer returning an error or a list of tokens together with a list of warnings
- **addPos** is used to trace the position of the token
- **lexerError** to generate lexer error
- **lexerWarning** to generate lexer warnings
- **runLexerMonad** main function to handle the `LexerMonad` results

LexerMessage contains functions to handle errors and warnings from the lexer. Based on the warning/error type and the needed language, `LexerMessage` will fetch the correct description of an error or a warning out of the `LexerTexts` module. The show functions for the error and warning are maintained in this module.

LexerTexts fetches the correct description of an error or a warning out of the `LexerTexts` module. This centralization provides an easy entry point for the maintenance of the actual messages as these messages are no longer dispersed over the module functions.

LexerBinaryTrees module responsible for searching binary trees in an efficient way, to support the token recognition. This is the previously existing `UU_BinaryTrees` module which is renamed to match the used naming structure of the new lexer modules.

LexerToken contains the data structure, and corresponding show function, that represents the input tokens for the lexer.

3.2.3 New token structure

Based on the improvement topics mentioned in subsection 2.4, a new token structure is defined. Each token contains the lexeme: a part of the input string defining the token type and content, plus the position of the token in the input file. The token structure is defined as follows:

```
data Token = Tok { tokLex :: Lexeme
                 , tokPos :: FilePos
                 }
```

```
data Lexeme  = LexSymbol      Char
             | LexOperator    String
             | LexKeyword     String
             | LexString      String
             | LexExpl        String
             | LexAtom        String
             | LexDecimal     Int
             | LexOctal       Int
             | LexHex         Int
             | LexConId       String
             | LexVarId       String
  deriving (Eq, Ord)
```

`Lexeme` is the combination of the token type and the actual token content, sliced from the input string. `FilePos` is used to keep track of the original position of the lexeme in the input string.

During the lexer processing, the input file is processed sequentially. All kinds of different accepted constructions are checked in a specific order, and each time a match is found, the lexeme is extracted from the input string and the token is created. In the token creation (function `returnToken`), the position and the lexeme are grouped into a token, then the next lexer iteration is started.

3.3 Parser (R-M)

The mainstream design of the new parser has not changed much. Basically, each EBNF rule receives its own parser function. Thanks to the combinator operators, each parsing function also looks very similar to its corresponding EBNF.

The applicative interface is consistently used. By changing details of the implementation, e.g. the order of the fields in the parse tree, we have made many of the 'rebuild' functions unnecessary. For some parsers the amount of changes necessary in order to remove supporting functions was too large or even impossible with the current parse tree.

Note that in parts of the parser, the function syntax has substituted the record syntax for creating data objects. This was done only when the code readability could be improved by doing so.

3.3.1 Parsec

As mentioned earlier, and described in research context document [**?**], the new Ampersand parser has been rebuilt with another parsing library, namely Parsec. However, for the Ampersand developers, the source code of the parser will still look very familiar, thanks to the applicative interface. For developers, the main differences between Parsec and the uulib are:

- Parsec does not backtrack by default. In order to enable backtracking, the `try` function must be used. This is described in subsubsection 3.3.2.
- Parsec does not try to solve parsing errors. The parser stops immediately after the first issue. See also the error analysis in subsection 3.5.
- Error messages are customizable by using the `<?>` operator. This is also suggested in subsection 3.6.

- Some combinators have a different name, e.g. one must use `option` instead of `opt`. Assuming the documentation found on Hackage is clear and sufficient, interface differences are not documented here.

3.3.2 Backtracking

In order to explain the differences on backtracking behavior between the uulib and Parsec, we quote here Doaitse Swierstra, the author of the uulib [**?**]:

> *To understand the subtleties it is important to understand the differences between the try construct in Haskell and the non-greedy parsing strategy used in uu-parsinglib. Effectively the latter is a try which just looks ahead one symbol. In that respect it is less powerful than the try construct from Parsec, in which you specify that a specific construct has to be present completely. And then there is the underlying different overall strategy. Parsec uses a back-tracking strategy with explicit tries to commit, whereas uu-parsinglib uses a breadth-first strategy with an occasional single symbol look-ahead.*

We can therefore conclude that the try-statements in Parsec are undesirable. However, they are necessary when the grammar is ambiguous. In this section we explain why each of the remaining try statements are necessary, and how these issues can be resolved:

Classify This ambiguity in the grammar arises from the `Classify` and `GenDef` productions:

```
Classify ::= 'CLASSIFY' ConceptRef 'IS' Cterm
GenDef ::= ('CLASSIFY' | 'SPEC') ConceptRef 'ISA' ConceptRef
```

When the parser encounters `'CLASSIFY'`, it cannot define whether it found a `Classify` or a `GenDef` production. Therefore, the parser must consume the keyword and a `ConceptRef` before consuming either `'IS'` or `'ISA'` and determining which production is applicable.

In order to solve this issue, one must choose a different keyword or symbol for each of the productions. Another option would be to merge the two statements in the same parser. We did not merge the productions because that would make the parser less maintainable.

Role This ambiguity in the grammar arises from the `RoleRelation` and `RoleRule` productions:

```
RoleRelation ::= 'ROLE' RoleList 'EDITS' NamedRelList
RoleRule ::= 'ROLE' RoleList 'MAINTAINS' ADLidList
```

When the parser encounters `'ROLE'`, it cannot define whether it is a `RoleRelation` or a `RoleRule` production. Therefore, the parser must consume the keyword and a `RoleList` (which may be long) before consuming either `'MAINTAINS'` or `'EDITS'` and determining which production is applicable.

In order to solve this issue, one must choose a different keyword for each of the productions, merge the two options to have the same representation in the parse tree, or refactor the parser so that the two options are parsed together. We did not merge the productions because that would make the parser less maintainable.

View This ambiguity in the grammar arises from the `FancyViewDef` and `ViewDefLegacy` productions:

```
FancyViewDef ::= 'VIEW' Label ConceptOneRefPos 'DEFAULT'? '{' ViewObjList
'}' HtmlView?  'ENDVIEW'
ViewDefLegacy ::= ('VIEW' | 'KEY') LabelProps ConceptOneRefPos '('
ViewSegmentList ')'
```

When the parser encounters `'VIEW'`, it cannot define whether it found a `FancyViewDef` or a `ViewDefLegacy` production. In this case, defining which construction is applicable is even more complicated. This decision must, in the worst case, be delayed until the parser encounters a `'{'` or `'('`. That's because the productions `Label` and `LabelProps` are not disjoint, and `'DEFAULT'` is optional.

In order to solve this issue, we advise to merge or drop the legacy statement.

Multiplicity This ambiguity in the grammar arises from the `Mult` production:

```
Mult ::= ('0' | '1') '..'  ('1' | '*') | '*' | '1'
```

When the parser encounters `'1'`, it cannot define whether it found the first or the last production. The parser must therefore read the next token before choosing the right option.

In order to solve this issue, we advise to refactor the grammar (and the parser) to have the following production:

```
Mult ::= '0' '..'  ('1' | '*') | '1'('..'  ('1' | '*'))?  | '*'
```

We did not refactor the code in this manner because the `pMult` parser does more than only parsing: it also changes the representation of the found constructions before creating the parse tree.

Labels and Terms In the productions `Att` and `RuleDef`, we see very similar ambiguities:

```
Att ::= LabelProps?  Term
RuleDef ::= 'RULE' Label?  Rule Meaning* Message* Violation?
```

Wherein:

```
Label ::= ADLid ':'
LabelProps ::= ADLid ('' ADLidListList '')?  ':'
Rule ::= Term ('=' Term | '|-' Term)?
```

And one of the possible productions of `Term` is:

```
Term ::= Trm2 ::= Trm3 ::= Trm4 ::= Trm5 ::= Trm6 ::= RelationRef ::=
NamedRel ::= Varid Sign?
```

While:

```
ADLid ::= Varid | Conid | String
```

What happens here is that when the parser encounters a `Varid`, it cannot define whether it is part of the (optional) `Label` production or if no `Label` was given and the `Varid` is part of a `Term`/`NamedRel` production.

Due to the quite complex grammar for the `Term` production, this issue may severely impact the parser's performance. This is probably the most harmful of the ambiguities mentioned. However, it can only be solved by adding a symbol before the `Term` production (e.g. making the ':' non-optional).

Please note that in order to have proper backtracking with correct error messages, Parsec may require two try-statements [**?**].

3.4 Parse tree (R-M)

Improvements in the Ampersand parse tree are out of the scope of this project, because of the potential consequences to the rest of the Ampersand system. However, during the development of the new parser a few constructions have been changed in order to make the parser more readable and maintainable. The changes have been mostly in the order of the constructor parameters, and this was done consequently though all Ampersand modules. The updated parse tree is depicted in the appendices (Figure 5).

3.5 Errors

The proof of the pudding is in the eating, and therefore, our error message qualification definition from subsection 2.6 is applied on the new parser. All erroneous syntax statements recorded during the analysis phase are fed to the new parser to draw up the error message quality overview of the new parser.

The results are placed in Table 2 next to the results of the old parser as a reference point. The actual error list, containing the syntax statements with the corresponding error messages, is available in section 5.2.

Error quality	Old parser		New parser	
Good	19	22,35%	70	82,35%
Average	18	21,18%	14	16,47%
Bad	48	56,47%	1	1,18%
Total	**85**	**100,00%**	**85**	**100,00%**

Table 2: Error message comparison results

The table clearly visualizes that there indeed was an issue with the error messages generated by the old Ampersand parser. By implementing the new parser, an improvement of 360% is made towards the creation of good error messages in which 82% of the generated error messages are good, compared to 22% in the old Parser.

One bad error message remains in the system. After analysis, the resolution of this remaining bad error would require too much effort, hence increased complexity, compared to the actual value gain. The error in question is thrown when the `Meta` statement is used with only one string value. Given the simple structure of the notation, we assume that the reference to the line number of the `Meta` is sufficient although the error message is unclear. Therefore, this message is kept as is.

3.6 Next steps (M)

During the parser re-implementation and code re-factoring to enhance the code maintainability, we noticed potential improvement topics in Ampersand. Were possible, with respect to our project requirements and our milestones, we integrated these topics in the new parser. Some topics we could not handle due to time constraints or given the undesired impact on the surrounding modules. It would be a pity if these improvement ideas were lost as these can help the Ampersand team to further enhance the tool. This section summarizes our improvement suggestions, both generic as specific ones, for the Ampersand tool.

Syntax improvements In the syntax, we discovered some statements which are not pure LL(k) statements. The Parsec `try` function allows the backtracking in the parser, avoiding that

input is consumed which is still needed in another parse statements if the parse function can not succeed successfully. Using `try` we can handle this situation but the backtracking has a negative impact of the parser performance whilst it adds complexity to the parser module. Our suggestion is to further optimize the Ampersand syntax to establish a pure predictive syntax in which no backtracking is needed. This will not only improve the parser performance and maintainability, the syntax simplifications will . The syntax statements in which backtracking is needed, including the reasons why, are listed in section subsubsection 3.3.2.

Warnings An improvement point in the new lexer is that warnings are now supported. Warnings are however not yet integrated in the Ampersand tool. There is no need to stop the compiling process for warnings, as the reasons behind them can make sense. Warnings can, however, support the user identifying the cause of unexpected results, although the compilation could be completed successfully. Our suggestion is to add the list of warnings to the design artifacts of Ampersand, available for the user to reflect on is needed.

Uniform parse tree structure In the parse tree, not all data types are constructed in correspondence to the syntax of the Ampersand language. Several restructuring functions are used in the parser to reformulate the result of the parse functions to match the constructor type in the parse tree. These functions can be recognized in the parser as rebuild or reorder functions. An example of such a rebuild action is given below:

The syntax notation of `ViewAtt` is defined as: `ViewAtt ::= LabelProps?  Term`

The parser function will use the order of the notation to extract the needed information:

```
pViewAtt :: AmpParser P_ObjectDef
pViewAtt = rebuild <$> currPos <*> optLabelProps <*> pTerm
    where rebuild pos (nm, strs) ctx = P_Obj nm pos ctx mView msub strs
          mView = Nothing
          msub  = Nothing
```

When we look at the data type as defined in the parse tree, the following order is defined:

```
data P_ObjDef a =
        P_Obj { obj_nm :: String
                    , obj_pos :: Origin
                    , obj_ctx :: Term a
                    , obj_mView :: Maybe String
                    , obj_msub :: Maybe (P_SubIfc a)
                    , obj_strs :: [[String]]
                    }
```

As the order of the parameters in the parse tree and the parser are different, the intermediate function rebuild is used to align both types to each other. Our personal experience was that these rebuild functions make the parser code more difficult to understand and hence, to maintain. To further decrease the code complexity, we suggest to try to eliminate these rebuild functions by restructuring some data types in the parse tree.

Manual overrule of error message Our analysis of the new error messages showed that the quality of these improved distinctively. Parsec provides to possibility to overrule the standard Parsec error message by an own formulated message. During our implementation we decided to stick to the standard if the standard error message was, at least, sufficient. If the Ampersand teams want to tweak some error message after all, this can be realized using the `<?>` Parsec function. Placing the `<?>` after the parser, followed by a string, changes the standard Parsec text after the keyword 'expecting'.

4 Test Report

4.1 Test suite (R-M)

Together with the new parser, a test suite has been developed. This test suite has been used to verify the performance and correctness of the new parser. The source code can be found in the folder `src/Database/Design/Ampersand/Test` within the Ampersand repository.

The test suite runs in three steps, which are depicted in Figure 4. Each of the modules are described in the following subsection.

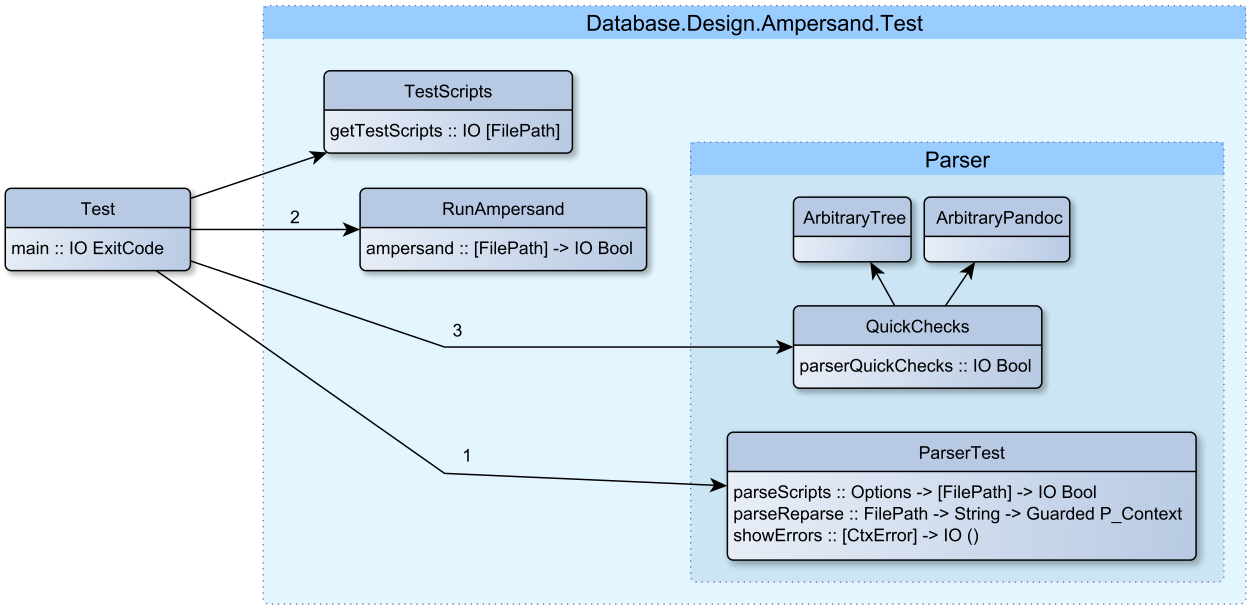


Figure 4: Test suite modules with their exported functions

4.1.1 Modules

In this section a short description of each module is given:

Test contains the `main` method that can be executed to run the test suite. The `main` function calls each of the other modules in sequence, stopping if any of them returns `False`. When all tests have been successful, the return code is `ExitSuccess`. Otherwise, the return code is naturally `ExitFailure`.

TestScripts retrieves a list of scripts that can be used for the different tests. It searches for tests within the folder `ArchitectureAndDesign`, and contains a list of scripts from the

`ampersand-models` repository, that can be changed at a later moment if wished. Note that all the ADL-scripts listed in this section must be correct for the parser and the type checker. During development, the list was limited to the scripts that could be successfully executed by the original ampersand version.

ParserTest exports three functions that are the core of testing the parser:

- `parseScripts` receives a list of files to parse, and checks that every file can be parsed successfully.
- `parseReparse` tries to parse a file, and if successful, pretty-prints the result and parses it again.
- `showErrors` prints the given parse errors to the output.

RunAmpersand receives a list of files, and checks that every file can be executed successfully by Ampersand. This tests thus not only the parser, but also the interface between the parser and the type checker, as the rest of the Ampersand chain.

QuickChecks generates random parse tree structures and generates the corresponding ADL-script by pretty printing the parse tree. This ADL-script is then fed back to the parser through the `parseReparse` function, to verify that the parser can accept any random input. More information on the quick checks is given in subsection 4.1.2.

ArbitraryTree is a support module that gives `Arbitrary` instances to all parse-tree structures. This is used by QuickCheck as described in subsection 4.1.2.

ArbitraryPandoc contains `Arbitrary` instances to the Pandoc data types. This file has not been developed in this project, but copied from the `jgm/pandoc` project with the GPL license.

4.1.2 QuickCheck and pretty printing

The most innovative part of the test suite is the use of random structures to test the parser. In this section we describe how this generation is implemented.

The main role in the generation of random structures is played by the support library QuickCheck, which has been added to the Ampersand project. QuickCheck is able to generate any data structure randomly. However, since the parse tree is a custom structure that must obey specific rules, QuickCheck requires the specification of these rules by instances of the `Arbitrary` class.

Every data structure in the parse tree has received an `Arbitrary` instance used for test purposes. The instances can be found in the module `Database.Design.Ampersand.Test.Parser-.ArbitraryTree`, as described in subsection 4.1.1.

After generating the random parse trees, the test suite needs to convert them to ADL-scripts. The conversion of parse tree to source code is also known as pretty printing. As the pretty printing is seen as part of the parse tree, it is not included in the Test modules, but is part of the input subsystem. The pretty printing instances are found in the module `Database.Design.Ampersand.ADL1.PrettyPrinters`. This module makes use of the library `Text.PrettyPrint.Leijen`, that outlines the output so it is indeed 'pretty'.

Now that the ADL source is available, the parser is executed. The result of the parser is checked to be equal to the generated tree by the property `prop_pretty`. The property is currently tested for 64 generated parse trees in the test suite. If the test fails for any generated structure, the test suite fails with an appropriate error.

4.1.3 Running the tests

During the parser development, the `main` function of the parser tests has been executed manually, through a batch file. This is mainly done because the project team did not have access to the Sentinel server, and no documentation was available on how to run Sentinel locally in a Windows machine. However, now that the parser is being delivered, it should be integrated with the other existing Ampersand/Sentinel tests. We leave the option open for the Ampersand development team to either add the Sentinel jobs to this test suite, or to add the parser test suite to the Sentinel jobs.

4.1.4 Test coverage (D)

TODO: HPC test coverage description.

4.2 Warnings (M)

TODO: Compare the compiler warnings before and after the refactoring.

4.3 Errors (M)

Since evaluating the quality of error messages is manual work, the errors have not been included in the test suite. TODO: Give Maarten's findings on how the errors have improved. Maybe the tables should be an attachment, but the summary should be here.

4.4 Next steps (R-M)

In this section we name a couple changes that can be done in the test suite in the future:

Sentinel During the development of the new parser, we worked in a separate fork. Our changes were not being tested in the Ampersand test server (Sentinel). Since we did not have access to this server, we developed a separate test suite. It may be pertinent to integrate the Sentinel jobs into the test suite or to integrate the test suite into the Sentinel jobs.

Output Currently, the test suite outputs errors by using the `Debug.Trace` module. From a purely functional perspective, using this module may be undesirable. Therefore, the Ampersand team may consider changing the test outputs to use IO with monads in a more functional way.

5 Conclusion

5.1 Achievements (M)

TODO: Conclude how the achievements will help the further development of Ampersand and its context.

5.2 Next steps (R-M)

Although we believe the new parser is a large step forward, we also recognize that there are still improvements to be made.

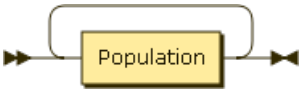
Ampersand is growing and changing in a fast pace. This is a direct consequence of a project involved with research and active development. In such projects, it is often impossible to predict which functionalities will be necessary and to design the domain-specific language accordingly beforehand.

As expected we see that most of the remaining issues are related to the grammar ambiguities that force backtracking. Also the parse tree is not consistently designed. These issues are mentioned in subsection 3.6. Other improvements are also possible in the test framework delivered. Those are mentioned in subsection 4.4.

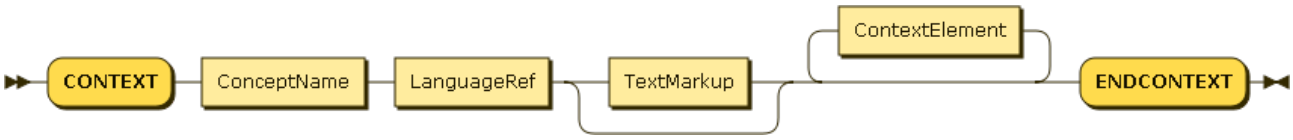
Appendices

EBNF Diagrams

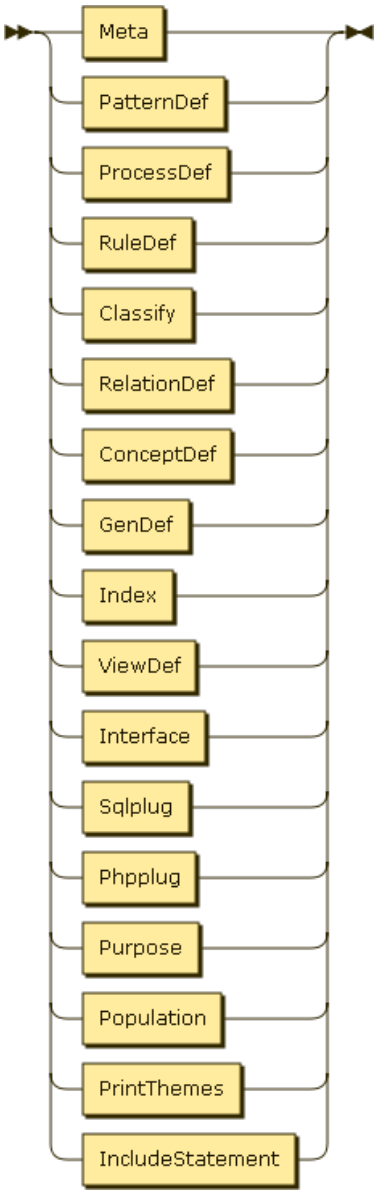
The diagrams below have been generated with the Railroad Diagram Generator (`http://bottlecaps.de/rr/ui`).



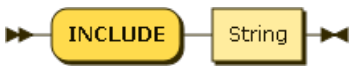

```
Populations ::= Population+
```



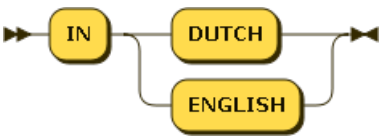

```
Context ::= 'CONTEXT' ConceptName LanguageRef TextMarkup?  ContextElement*
'ENDCONTEXT'
```



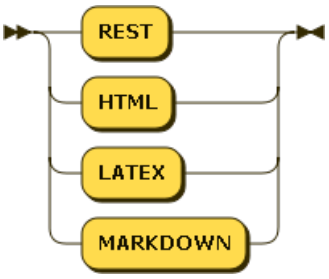
`ContextElement ::= Meta | PatternDef | ProcessDef | RuleDef | Classify | RelationDef | ConceptDef | GenDef | Index | ViewDef | Interface | Sqlplug | Phpplug | Purpose | Population | PrintThemes | IncludeStatement`



`IncludeStatement ::= 'INCLUDE' String`

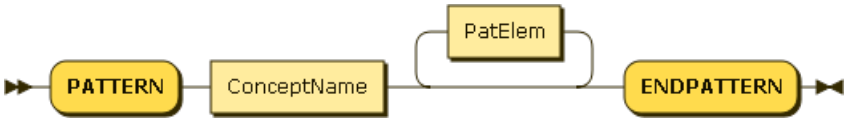


`LanguageRef ::= 'IN' ('DUTCH' | 'ENGLISH')`

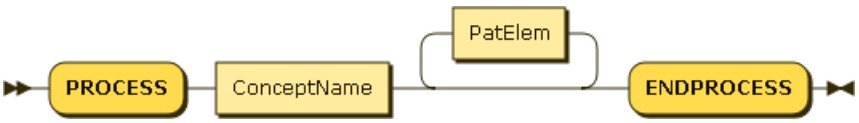


`TextMarkup ::= 'REST' | 'HTML' | 'LATEX' | 'MARKDOWN'`

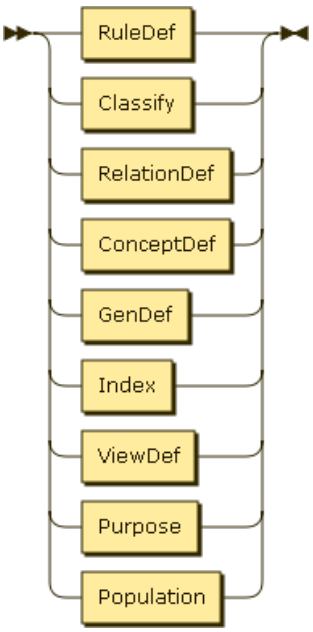
`Meta ::= 'META' String String`



`PatternDef ::= 'PATTERN' ConceptName PatElem* 'ENDPATTERN'`

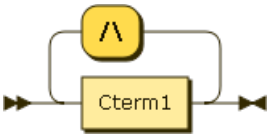


`ProcessDef ::= 'PROCESS' ConceptName PatElem* 'ENDPROCESS'`

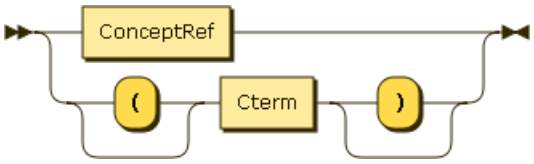


`PatElem ::= RuleDef | Classify | RelationDef | ConceptDef | GenDef | Index | ViewDef | Purpose | Population`

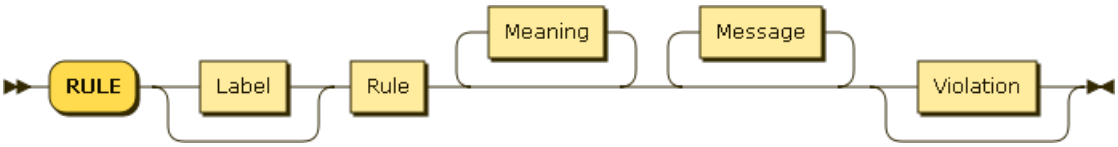
Classify ::= ‘CLASSIFY’ ConceptRef ‘IS’ Cterm



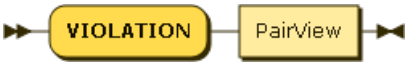
Cterm ::= Cterm1 (‘/\’ Cterm1)*



Cterm1 ::= ConceptRef | (‘(’? Cterm ‘)’?)

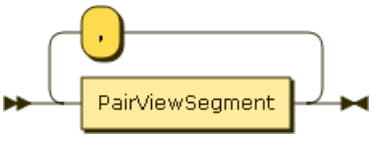


RuleDef ::= ‘RULE’ Label? Rule Meaning* Message* Violation?

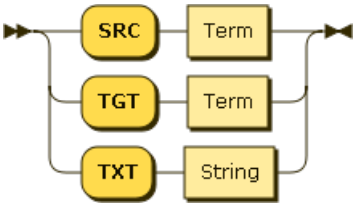


Violation ::= ‘VIOLATION’ PairView

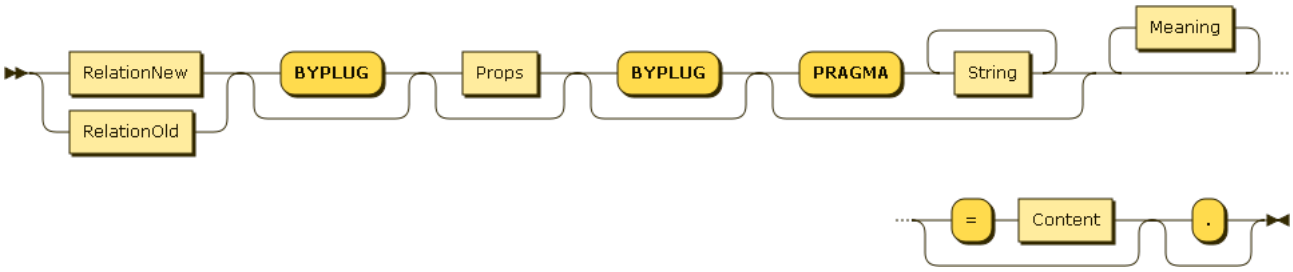
PairView ::= ‘(’ PairViewSegmentList ‘)’



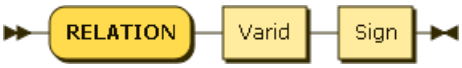
PairViewSegmentList ::= PairViewSegment (‘,’ PairViewSegment)*



PairViewSegment ::= ‘SRC’ Term | ‘TGT’ Term | ‘TXT’ String

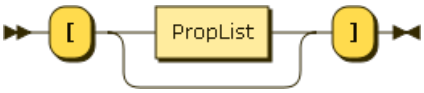


`RelationDef ::= (RelationNew | RelationOld) 'BYPLUG'? Props? 'BYPLUG'? ('PRAGMA' String+)? Meaning* ('=' Content)? '.'?`

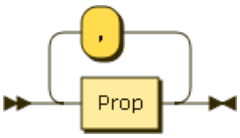


`RelationNew ::= 'RELATION' Varid Sign`

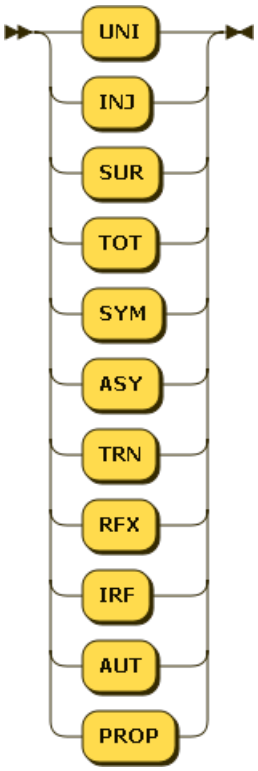
`RelationOld ::= Varid '::' ConceptRef Fun ConceptRef`



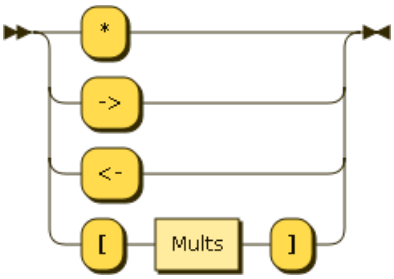
`Props ::= '[' PropList? ']'`



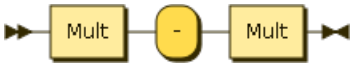
`PropList ::= Prop (',' Prop)*`



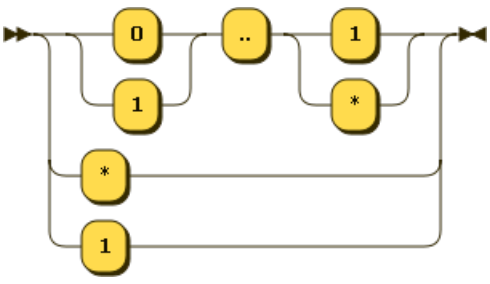
Prop ::= ‘UNI’ | ‘INJ’ | ‘SUR’ | ‘TOT’ | ‘SYM’ | ‘ASY’ | ‘TRN’ | ‘RFX’ | ‘IRF’ | ‘AUT’ | ‘PROP’



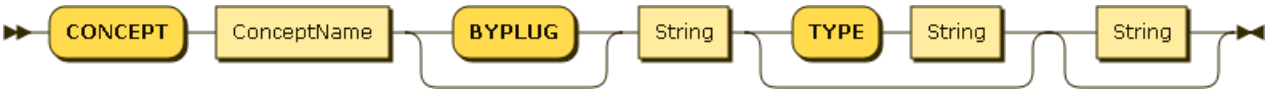
Fun ::= ‘*’ | ‘->’ | ‘<-’ | ‘[’ Mults ‘]’



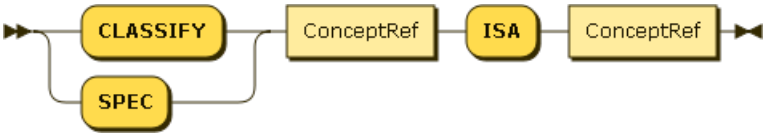
Mults ::= Mult ‘-’ Mult



Mult ::= (‘0’ | ‘1’) ‘..’ (‘1’ | ‘*’) | ‘*’ | ‘1’

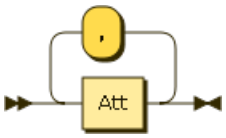


`ConceptDef ::= 'CONCEPT' ConceptName 'BYPLUG'? String ('TYPE' String)? String?`

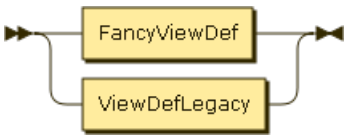


`GenDef ::= ('CLASSIFY' | 'SPEC') ConceptRef 'ISA' ConceptRef`

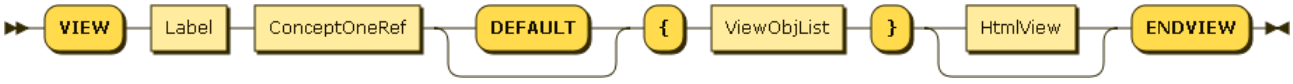
`Index ::= 'IDENT' Label ConceptRef '(' IndSegmentList ')'`



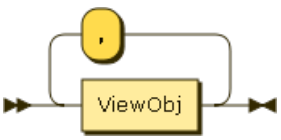
`IndSegmentList ::= Att (',' Att)*`



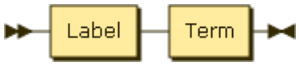
`ViewDef ::= FancyViewDef | ViewDefLegacy`



`FancyViewDef ::= 'VIEW' Label ConceptOneRef 'DEFAULT'? '{' ViewObjList '}' HtmlView? 'ENDVIEW'`

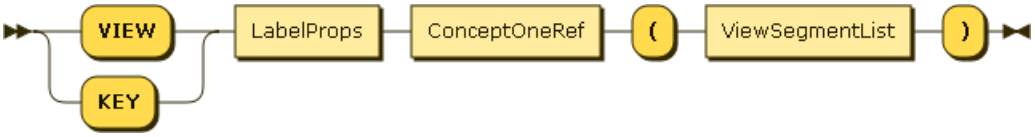


`ViewObjList ::= ViewObj (',' ViewObj)*`

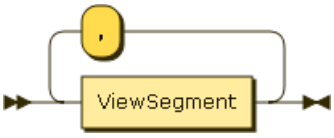


`ViewObj ::= Label Term`

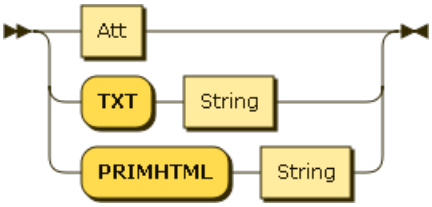
HtmlView ::= 'HTML' 'TEMPLATE' String



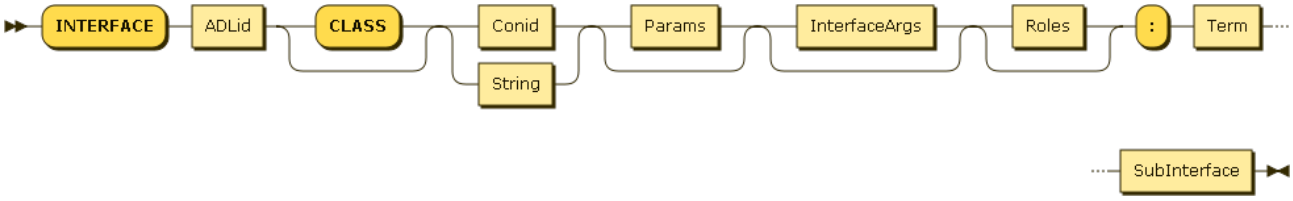
ViewDefLegacy ::= ('VIEW' | 'KEY') LabelProps ConceptOneRef '(' ViewSegmentList ')'



ViewSegmentList ::= ViewSegment (',' ViewSegment)*



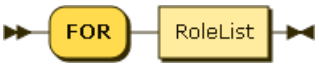
ViewSegment ::= Att | 'TXT' String | 'PRIMHTML' String



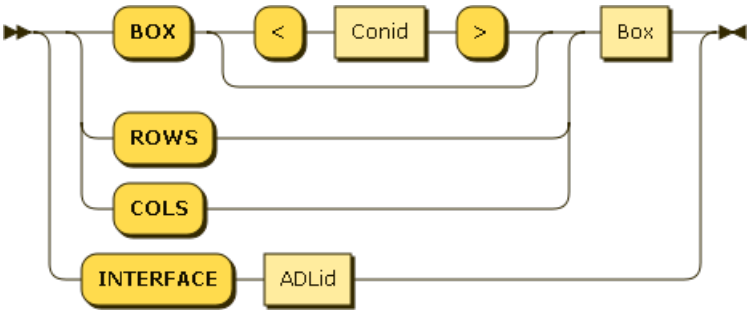
Interface ::= 'INTERFACE' ADLid 'CLASS'? (Conid | String) Params? InterfaceArgs? Roles? ':' Term SubInterface

Params ::= '(' NamedRel ')'

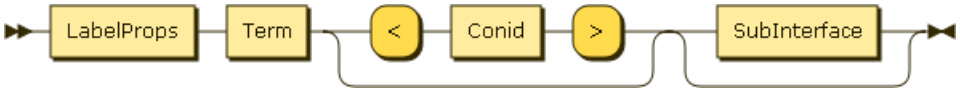
InterfaceArgs ::= '{' ADLidListList '}'



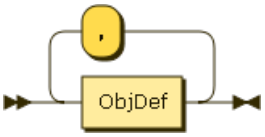
Roles ::= ‘FOR’ RoleList



SubInterface ::= (‘BOX’ (‘<’ Conid ‘>’)? | ‘ROWS’ | ‘COLS’) Box | ‘INTERFACE’ ADLid



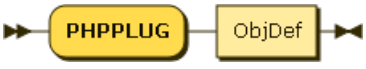
ObjDef ::= LabelProps Term (‘<’ Conid ‘>’)? SubInterface?



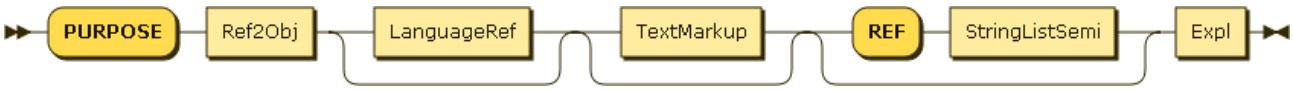
ObjDefList ::= ObjDef (‘,’ ObjDef)*

Box ::= ‘[’ ObjDefList ‘]’

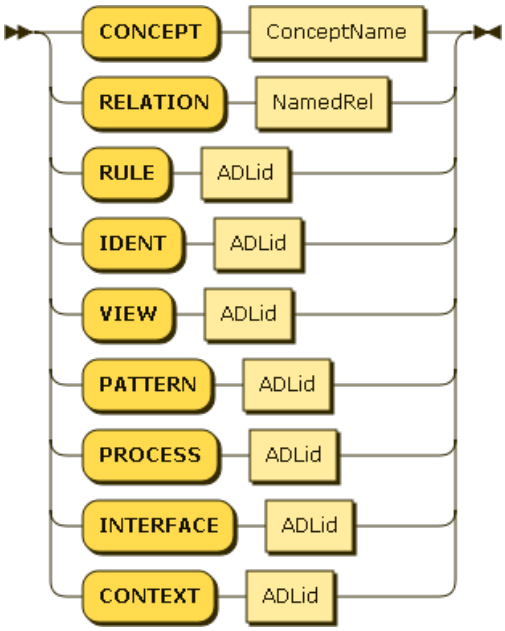
Sqlplug ::= ‘SQLPLUG’ ObjDef



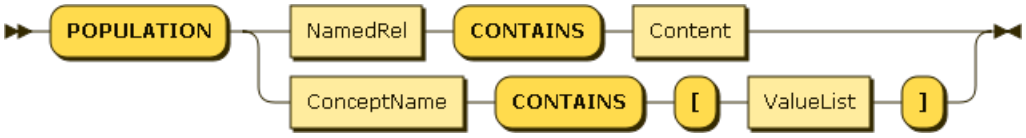
Phpplug ::= ‘PHPPLUG’ ObjDef



Purpose ::= ‘PURPOSE’ Ref2Obj LanguageRef? TextMarkup? (‘REF’ StringListSemi)? Expl



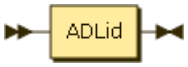
`Ref2Obj ::= 'CONCEPT' ConceptName | 'RELATION' NamedRel | 'RULE' ADLid | 'IDENT' ADLid | 'VIEW' ADLid | 'PATTERN' ADLid | 'PROCESS' ADLid | 'INTERFACE' ADLid | 'CONTEXT' ADLid`



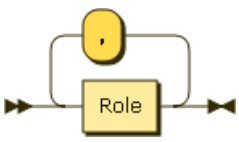
`Population ::= 'POPULATION' (NamedRel 'CONTAINS' Content | ConceptName 'CONTAINS' '[' ValueList ']')`

`RoleRelation ::= 'ROLE' RoleList 'EDITS' NamedRelList`

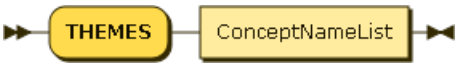
`RoleRule ::= 'ROLE' RoleList 'MAINTAINS' ADLidList`



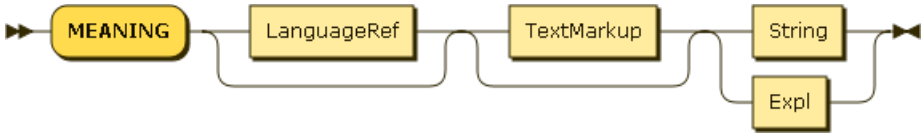
`Role ::= ADLid`



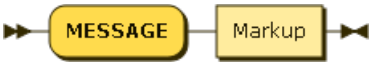
`RoleList ::= Role (',' Role)*`



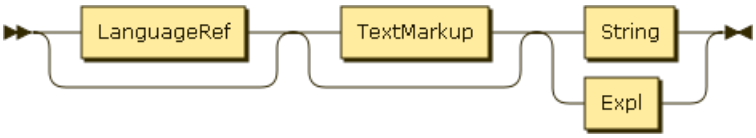
`PrintThemes ::= 'THEMES' ConceptNameList`



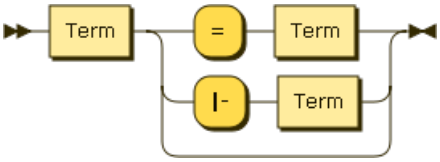
`Meaning ::= 'MEANING' LanguageRef? TextMarkup? (String | Expl)`



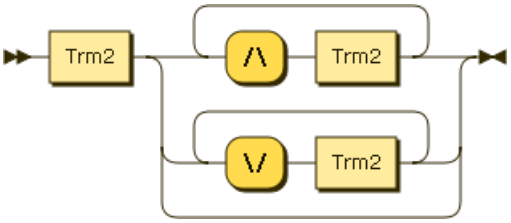
`Message ::= 'MESSAGE' Markup`



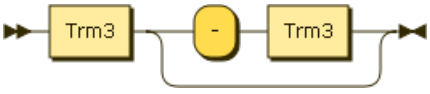
`Markup ::= LanguageRef? TextMarkup? (String | Expl)`



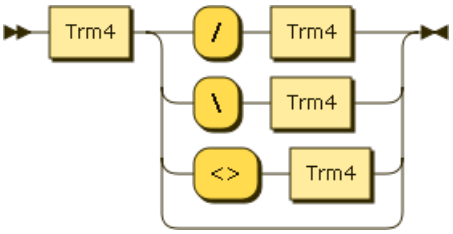
`Rule ::= Term ('=' Term | '|-' Term)?`



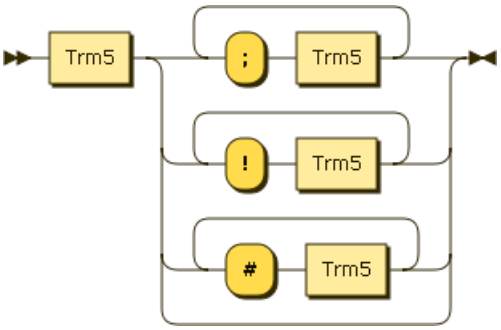
`Term ::= Trm2 (('/\' Trm2)+ | ('\/' Trm2)+)?`



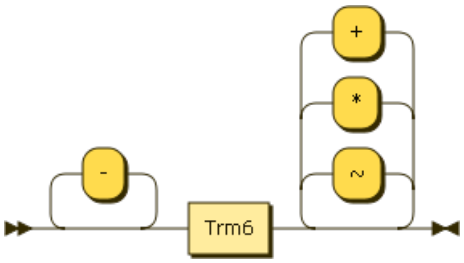
`Trm2 ::= Trm3 ('-' Trm3)?`



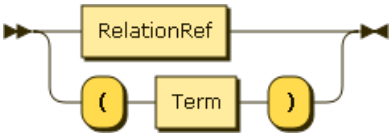
`Trm3 ::= Trm4 (‘/’ Trm4 | ‘\’ Trm4 | ‘<>’ Trm4)?`



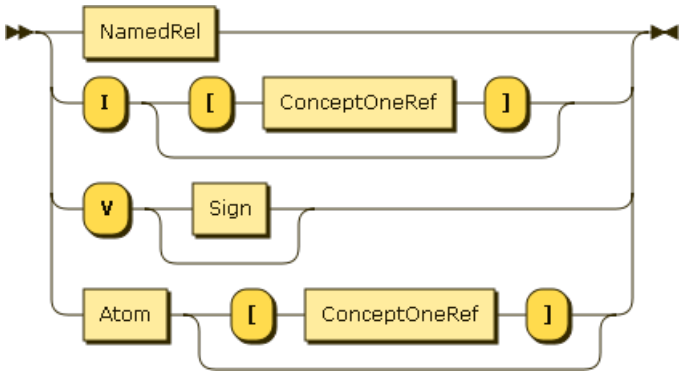
`Trm4 ::= Trm5 ((‘;’ Trm5)+ | (‘!’  Trm5)+ | (‘#’ Trm5)+)?`



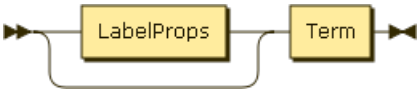
`Trm5 ::= ‘-’* Trm6 (‘ ’ | ‘*’ | ‘+’)*`



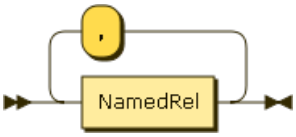
`Trm6 ::= RelationRef | ‘(’ Term ‘)’`



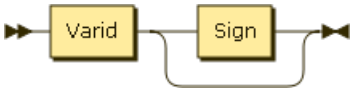
`RelationRef ::= NamedRel | 'I' ('[' ConceptOneRef ']')? | 'V' Sign? | Atom ('[' ConceptOneRef ']')?`



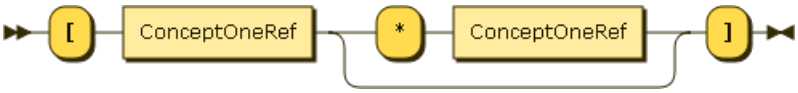
`Att ::= LabelProps? Term`



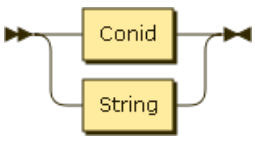
`NamedRelList ::= NamedRel (',' NamedRel)*`



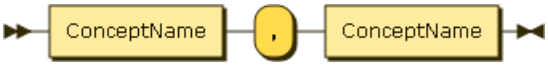
`NamedRel ::= Varid Sign?`



`Sign ::= '[' ConceptOneRef ('*' ConceptOneRef)? ']'`

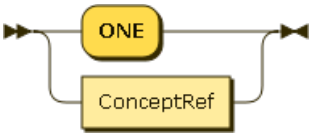


`ConceptName ::= Conid | String`



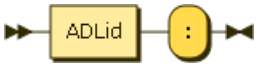
`ConceptNameList ::= ConceptName (',' ConceptName)`

`ConceptRef ::= ConceptName`

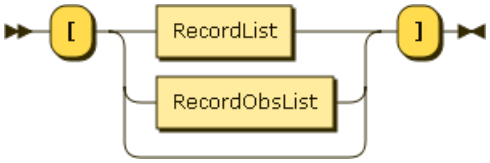


`ConceptOneRef ::= 'ONE' | ConceptRef`

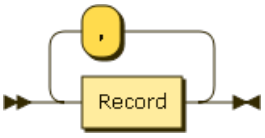
`LabelProps ::= ADLid ('{' ADLidListList '}')? ':'`



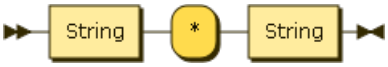
`Label ::= ADLid ':'`



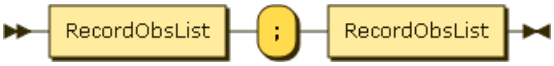
`Content ::= '[' (RecordList | RecordObsList)? ']'`



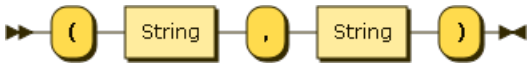
`RecordList ::= Record (',' Record)*`



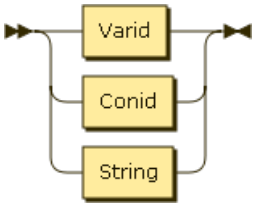
`Record ::= String '*' String`



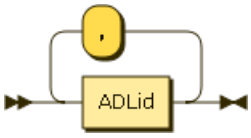
`RecordObsList ::= RecordObsList (';' RecordObsList)`



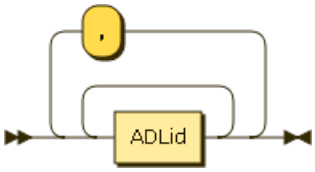
`RecordObs ::= '(' String ',' String ')'`



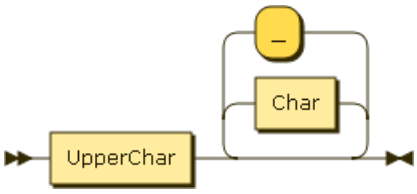
`ADLid ::= Varid | Conid | String`



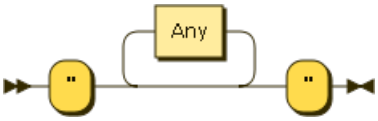
`ADLidList ::= ADLid (',' ADLid)*`



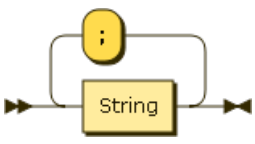
`ADLidListList ::= ADLid+ (',' ADLid+)*`



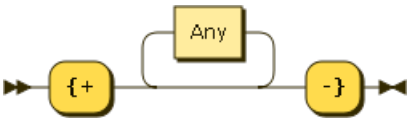
`Conid ::= UpperChar (Char | '_')*`



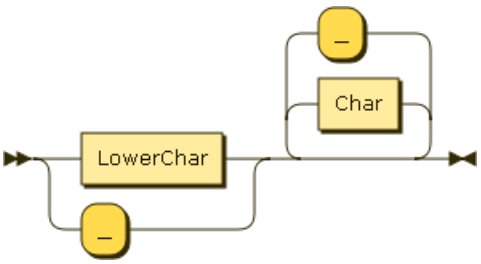
`String ::= '"' Any* '"'`



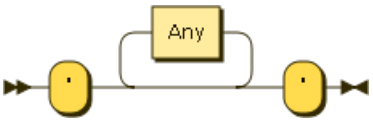
`StringListSemi ::= String (';' String)*`



`Expl ::= ‘{+’ Any* ‘-}’`



`Varid ::= (LowerChar | ‘_’) (Char | ‘_’)*`



`Atom ::= "‘" Any* "’"`

Parse Tree

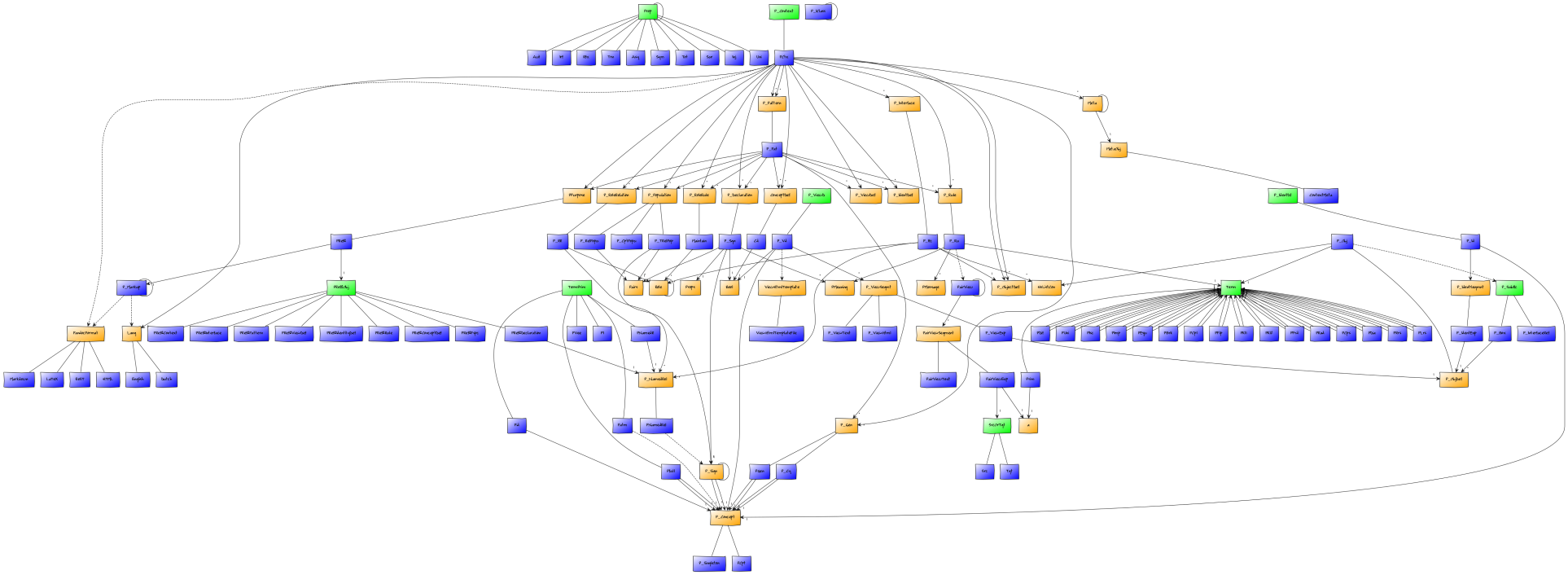


Figure 5: Diagram of the Ampersand parse tree. Connections without an arrow represent the data contructors. Connections with an arrow also show the multiplicity in the relationship (i.e. 1 or *) or are stripped to represent an optional relationship.

Error list (M)

Code	Old error	Quality	New error	New quality
use of 'CORNCEPT' instead of CONCEPT	Error(s) found: before upper case identifier CORNCEPT at line 7, column 1 of file "Hypotheek.adl " Expecting "ENDPATTERN" or (lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RU LE" or "SPEC" or "VIEW" ...)* Try inserting symbol "CONCEPT"	Good	[PE "ArchitectureAndDesign/Syntax/Hypotheek.adl" (line 7, column 1): unexpected Upper case identifier CORNCEPT expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"]	Yes

Code	Old error	Quality	New error	New quality
clientName :: Client -Z> Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "-" at line 14, column 22 of file "testfle_mba.adl" Expecting symbol [or "*" or "->" or "<-" Try inserting symbol symbol [========== before upper case identifier Z at line 14, column 23 of file "testfle_mba.adl" Expecting symbol] or "*" or "0" or "1" Try inserting symbol symbol] ========== before ">" at line 14, column 24 of file "testfle_mba.adl" Expecting symbol [or lower case identifier ?lc? or "." or "=" or "BYPLUG" or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PRAGMA" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "V IEW" or ("MEANING" ...)* Try deleting symbol ">" at line 14, column 24 of file "testfile_mba.adl" ========== before upper case identifier Name	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 22): unexpected Operator '-' expecting Operator '*', Operator '->', Operator '<-' or Symbol '['	Nice

Code	Old error	Quality	New error	New quality
clientName :: Client ** Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "*" at line 14, column 23 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try deleting symbol "*" at line 14, column 23 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 23): unexpected Operator '*'	Nice
clientName :: Client [] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before symbol] at line 14, column 23 of file "testfile_mba.adl" Expecting "*" or "-" or "0" or "1" Try inserting symbol "-"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 23): unexpected Symbol ']' expecting Integer 0, Hexadecimal 0x, Octal 0o, Integer 1, Hexadecimal 0x1, Octal 0o1, Operator '*' or Operator '-'	Nice
clientName :: Client [0..2-*] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before decimal Integer 2 at line 14, column 26 of file "testfile_mba.adl" Expecting "*" or "1" Try deleting symbol decimal Integer 2 at line 14, column 26 of file "testfile_mb a.adl" ========== before "-" at line 14, column 27 of file "testfile_mba.adl" Expecting "*" or "1" Try inserting symbol "1"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 26): unexpected Integer 2 expecting Integer 1, Hexadecimal 0x1, Octal 0o1 or Operator '*'	Nice

Code	Old error	Quality	New error	New quality
clientName :: Client [0..1] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before symbol] at line 14, column 27 of file "testfile_mba.adl" Expecting "-" Try inserting symbol "-"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 27): unexpected Symbol ']' expecting Operator '-'	
clientName :: Client [0-1] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "-" at line 14, column 24 of file "testfile_mba.adl" Expecting ".." Try inserting symbol ".." ========== before "-" at line 14, column 24 of file "testfile_mba.adl" Expecting "*" or "1" Try inserting symbol "1"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 24): unexpected Operator '-' expecting Operator '..'	

Code	Old error	Quality	New error	New quality
clientName :: Client [0-a] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "-" at line 14, column 24 of file "testfile_mba.adl" Expecting ".." Try inserting symbol ".." ========== before "-" at line 14, column 24 of file "testfile_mba.adl" Expecting "*" or "1" Try inserting symbol "1" ========== before lower case identifier a at line 14, column 25 of file "testfile_mba.adl" Expecting symbol] or "*" or "0" or "1" Try deleting symbol lower case identifier a at line 14, column 25 of file "testf ile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 24): unexpected Operator '-' expecting Operator '..'	

Code	Old error	Quality	New error	New quality
clientName :: Client [*-1..0] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "0" at line 14, column 28 of file "testfile_mba.adl" Expecting "*" or "1" Try deleting symbol "0" at line 14, column 28 of file "testfile_mba.adl" ========== before symbol] at line 14, column 29 of file "testfile_mba.adl" Expecting "*" or "1" Try inserting symbol "1"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 28): unexpected Integer 0 expecting Integer 1, Hexadecimal 0x1, Octal 0o1 or Operator '*'	
clientName :: 0Client [*-1..1] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before "0" at line 14, column 15 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try deleting symbol "0" at line 14, column 15 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 15): unexpected Integer 0	

Code	Old error	Quality	New error	New quality
clientName : Client [*-1..1] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before ":" at line 14, column 12 of file "testfile_mba.adl" Expecting "::" Try deleting symbol ":" at line 14, column 12 of file "testfile_mba.adl" ========== before upper case identifier Client at line 14, column 14 of file "testfile_mba.adl" Expecting "::" Try inserting symbol "::"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 12): unexpected Operator ':' expecting Operator '::'	

Code	Old error	Quality	New error	New quality
RELATION RelTest [RelationCR]	Error(s) found: before upper case identifier RelTest at line 29, column 10 of file "testfile_mba .adl" Expecting lower case identifier ?lc? Try deleting symbol upper case identifier RelTest at line 29, column 10 of file "testfile_mba.adl" ========== before symbol [at line 29, column 18 of file "testfile_mba.adl" Expecting lower case identifier ?lc? Try inserting symbol lower case identifier ?lc?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 29, column 10): unexpected Upper case identifier RelTest	

Code	Old error	Quality	New error	New quality
RELATION relTest [relationCR]	Error(s) found: before lower case identifier relationCR at line 29, column 19 of file "testfile_ mba.adl" Expecting upper case identifier ?uc? or "ONE" or string "" Try deleting symbol lower case identifier relationCR at line 29, column 19 of fi le "testfile_mba.adl" ========== before symbol] at line 29, column 29 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or "ONE" or string "" Try inserting symbol "ONE"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 29, column 19): unexpected Lower case identifier relationCR expecting Keyword "ONE"	
RELATION relTest [ONNE]	warning useful?			

Code	Old error	Quality	New error	New quality
RELATION relTest [ONE-ONNE]	Error(s) found: before "-" at line 19, column 22 of file "testfile_mba.adl" Expecting symbol] or "*" Try deleting symbol "-" at line 19, column 22 of file "testfile_mba.adl" ========== before upper case identifier ONNE at line 19, column 23 of file "testfile_mba.ad l" Expecting symbol] Try deleting symbol upper case identifier ONNE at line 19, column 23 of file "te stfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 19, column 22): unexpected Operator '-' expecting Operator '*' or Symbol ']'	

Code	Old error	Quality	New error	New quality
clientName :: client [*-1..1] Name = [("Client_1" , "Martijn") ; ("Client_2" , "Stef")]	Error(s) found: before lower case identifier client at line 27, column 15 of file "testfile_mba. adl" Expecting upper case identifier ?uc? or string "" Try deleting symbol lower case identifier client at line 27, column 15 of file " testfile_mba.adl" ========== before symbol [at line 27, column 22 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try inserting symbol upper case identifier ?uc?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 15): unexpected Lower case identifier client	

Code	Old error	Quality	New error	New quality
"clientName :: client [*-1..1] Name = [(""Client_1"" , ""Martijn"") ; (""Client_2"" , ""Stef"")]"	Error(s) found: before error in scanner: Unterminated string literal at line 27, column 1 of fil e "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol error in scanner: Unterminated string literal at line 27, co lumn 1 of file "testfile_mba.adl" ========== before symbol [at line 28, column 5 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol symbol [at line 28, column 5 of file "testfile_mba.adl"	Good	PE Lexer error LexerError line 27:1, file ArchitectureAndDesign/Syntax/testfile_mba.adl (NonTerminatedChar (Just "clientName :: client [*-1..1] Name =\r"))	

Code	Old error	Quality	New error	New quality
RELATION relTest [ONE*ONE]]	Error(s) found: before symbol] at line 19, column 27 of file "testfile_mba.adl" Expecting symbol [or lower case identifier ?lc? or "." or "=" or "BYPLUG" or "C LASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PRAGMA" or "PROCES S" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "V IEW" or ("MEANING" ...)* Try deleting symbol symbol] at line 19, column 27 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 19, column 27): unexpected Symbol ']' expecting Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	
RELATION relTest [[ONE*ONE]	Error(s) found: before symbol [at line 19, column 19 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or "ONE" or string "" Try deleting symbol symbol [at line 19, column 19 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 19, column 19): unexpected Symbol '[' expecting Keyword "ONE"	
VIEW	Not yet in ebnf			

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple IN ENGLISH REST2	Error(s) found: before upper case identifier REST2 at line 1, column 35 of file "testfile_mba.ad l" Expecting "ENDCONTEXT" or "HTML" or "LATEX" or "MARKDOWN" or "REST" or (lower ca se identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "INCLUDE" or "INTERF ACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" o r "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" ...)* Try deleting symbol upper case identifier REST2 at line 1, column 35 of file "te stfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 35): unexpected Upper case identifier REST2 expecting Keyword "REST", Keyword "HTML", Keyword "LATEX", Keyword "MARKDOWN", Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple REST2 IN ENGLISH	Error(s) found: before upper case identifier REST2 at line 1, column 24 of file "testfile_mba.ad l" Expecting "ENDCONTEXT" or "HTML" or "IN" or "LATEX" or "MARKDOWN" or "REST" or (lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PR OCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" o r "VIEW" ...)* Try deleting symbol upper case identifier REST2 at line 1, column 24 of file "te stfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 24): unexpected Upper case identifier REST2 expecting Keyword "IN"	

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple IN ENGLISHD	Error(s) found: before upper case identifier ENGLISHD at line 1, column 27 of file "testfile_mba .adl" Expecting "DUTCH" or "ENGLISH" Try deleting symbol upper case identifier ENGLISHD at line 1, column 27 of file "testfile_mba.adl" ========== before lower case identifier clientName at line 14, column 1 of file "testfile_m ba.adl" Expecting "DUTCH" or "ENGLISH" Try inserting symbol "DUTCH"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 27): unexpected Upper case identifier ENGLISHD expecting Keyword "DUTCH" or Keyword "ENGLISH"	

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple in ENGLISH	Error(s) found: before "ENGLISH" at line 1, column 27 of file "testfile_mba.adl" Expecting "::" Try deleting symbol "ENGLISH" at line 1, column 27 of file "testfile_mba.adl" ========== before lower case identifier clientName at line 14, column 1 of file "testfile_m ba.adl" Expecting "::" Try deleting symbol lower case identifier clientName at line 14, column 1 of fil e "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 24): unexpected Lower case identifier in expecting Keyword "IN"	

Code	Old error	Quality	New error	New quality
CONTEXT deliverySimple IN ENGLISH	Error(s) found: before lower case identifier deliverySimple at line 1, column 9 of file "testfil e_mba.adl" Expecting upper case identifier ?uc? or string "" Try deleting symbol lower case identifier deliverySimple at line 1, column 9 of file "testfile_mba.adl" ========== before "IN" at line 1, column 24 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try inserting symbol upper case identifier ?uc?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 9): unexpected Lower case identifier deliverySimple	

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple INCLUDE OFILEPATH IN ENGLISH	Error(s) found: before "0" at line 1, column 32 of file "testfile_mba.adl" Expecting string "" Try deleting symbol "0" at line 1, column 32 of file "testfile_mba.adl" ========== before upper case identifier FILEPATH at line 1, column 33 of file "testfile_mba .adl" Expecting string "" Try inserting symbol string "" ========== before upper case identifier FILEPATH at line 1, column 33 of file "testfile_mba .adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol "PURPOSE" ========== before upper case identifier	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 24): unexpected Keyword "INCLUDE" expecting Keyword "IN"	

Code	Old error	Quality	New error	New quality
CONTEXT DeliverySimple IN ENGLISH INCLUDE 0FILEPATH	Error(s) found: before "0" at line 1, column 43 of file "testfile_mba.adl" Expecting string "" Try deleting symbol "0" at line 1, column 43 of file "testfile_mba.adl" ========== before upper case identifier FILEPATH at line 1, column 44 of file "testfile_mba .adl" Expecting string "" Try deleting symbol upper case identifier FILEPATH at line 1, column 44 of file "testfile_mba.adl" ========== before lower case identifier clientName at line 14, column 1 of file "testfile_m ba.adl" Expecting string "" Try inserting symbol string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 1, column 43): unexpected Integer 0	

Code	Old error	Quality	New error	New quality
CONTEXT ENDCONTEXT3	Error(s) found: before upper case identifier ENDCONTEXT3 at line 237, column 1 of file "testfile _mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol upper case identifier ENDCONTEXT3 at line 237, column 1 of f ile "testfile_mba.adl" ========== unexpected end of input Expecting "ENDCONTEXT" Try inserting symbol "ENDCONTEXT"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 237, column 1): unexpected Upper case identifier ENDCONTEXT3 expecting Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	

Code	Old error	Quality	New error	New quality
META "authors" "Jan" "Paul"	Error(s) found: before string "Paul" at line 11, column 22 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol string "Paul" at line 11, column 22 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 11, column 22): unexpected String "Paul" expecting Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	
META "authors"	Error(s) found: before lower case identifier clientName at line 17, column 1 of file "testfile_mba.adl" Expecting string "" Try inserting symbol string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 17, column 1): unexpected Lower case identifier clientName	

Code	Old error	Quality	New error	New quality
METZA "authors" "Jan"	Error(s) found: before upper case identifier METZA at line 11, column 1 of file "testfile_mba.ad l" Expecting "ENDCONTEXT" or "HTML" or "LATEX" or "MARKDOWN" or "REST" or (lower ca se identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "INCLUDE" or "INTERF ACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" o r "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" ...)* Try inserting symbol "CONCEPT"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 11, column 1): unexpected Upper case identifier METZA expecting Keyword "REST", Keyword "HTML", Keyword "LATEX", Keyword "MARKDOWN", Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	

Code	Old error	Quality	New error	New quality
METZA "authors" "Jan""	Error(s) found: before error in scanner: Unterminated string literal at line 11, column 22 of fi le "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol error in scanner: Unterminated string literal at line 11, co lumn 22 of file "testfile_mba.adl"before error in scanner: Unterminated string literal at line 11, column 19 of fi le "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol error in scanner: Unterminated string literal at line 11, co lumn 19 of file "testfile_mba.adl"	Good	PE Lexer error LexerError line 11:22, file ArchitectureAndDesign/Syntax/testfile_mba.adl (NonTerminatedChar (Just "\r"))	

Code	Old error	Quality	New error	New quality
PATTEZRN Arbeidsduur -[Arbeidsrelaties]- werkgever :: Arbeidsrelatie -> Persoon PRAGMA "In" "is" "de werkgever" MEANING "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." PURPOSE RELATION werkgever {+Om de werkgever te kunnen bepalen gaan we ervan uit dat elke arbeidsrelatie precies n werkgever heeft.-} ENDPATTERN	Error(s) found: before upper case identifier PATTEZRN at line 10, column 1 of file "testfile_mba .adl" Expecting "ENDCONTEXT" or "HTML" or "LATEX" or "MARKDOWN" or "REST" or (lower ca se identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "INCLUDE" or "INTERF ACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" o r "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" ...)* Try deleting symbol upper case identifier PATTEZRN at line 10, column 1 of file "testfile_mba.adl" ========== before upper case identifier Arbeidsduur at line 10, column 10 of file "testfile _mba.adl" Expecting "ENDCONTEXT" or (lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPP LUG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 10, column 1): unexpected Upper case identifier PATTEZRN expecting Keyword "REST", Keyword "HTML", Keyword "LATEX", Keyword "MARKDOWN", Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	

Code	Old error	Quality	New error	New quality
BYPLUG [UNIS] BYPLUG	Error(s) found: before upper case identifier UNIS at line 14, column 9 of file "testfile_mba.adl " Expecting symbol] or ("ASY" or "INJ" or "IRF" or "PROP" or "RFX" or "SUR" or "S YM" or "TOT" or "TRN" or "UNI" ...)* Try deleting symbol upper case identifier UNIS at line 14, column 9 of file "tes tfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 9): unexpected Upper case identifier UNIS expecting Keyword "UNI", Keyword "INJ", Keyword "SUR", Keyword "TOT", Keyword "SYM", Keyword "ASY", Keyword "TRN", Keyword "RFX", Keyword "IRF", Keyword "AUT", Keyword "PROP" or Symbol ']'	
BYPLUG [UNI, TRN,] BYPLUG	Error(s) found: before symbol] at line 14, column 19 of file "testfile_mba.adl" Expecting "ASY" or "INJ" or "IRF" or "PROP" or "RFX" or "SUR" or "SYM" or "TOT" or "TRN" or "UNI" Try inserting symbol "UNI"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 19): unexpected Symbol ']' expecting Keyword "UNI", Keyword "INJ", Keyword "SUR", Keyword "TOT", Keyword "SYM", Keyword "ASY", Keyword "TRN", Keyword "RFX", Keyword "IRF", Keyword "AUT" or Keyword "PROP"	

Code	Old error	Quality	New error	New quality
BYPLUG [UNI, TRN] BYPLUGS	Error(s) found: before upper case identifier BYPLUGS at line 14, column 19 of file "testfile_mba .adl" Expecting lower case identifier ?lc? or "." or "=" or "BYPLUG" or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PRAGMA" or "PU RPOSE" or "RELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" or ("MEANING" ...)* Try deleting symbol upper case identifier BYPLUGS at line 14, column 19 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 14, column 19): unexpected Upper case identifier BYPLUGS expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
PRAGMA "In" d	Error(s) found: before "MEANING" at line 16, column 1 of file "testfile_mba.adl" Expecting "::" Try deleting symbol "MEANING" at line 16, column 1 of file "testfile_mba.adl" ========== before string "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de ro l van werkgever vervult." at line 16, column 9 of file "testfile_mba.adl" Expecting "::" Try inserting symbol "::" ========== before "PURPOSE" at line 17, column 1 of file "testfile_mba.adl" Expecting symbol [or "*" or "->" or "<-" Try inserting symbol "*" ========== before "PURPOSE" at line 17, column 1 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try inserting symbol upper case identifier ?uc?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 16, column 1): unexpected Keyword "MEANING" expecting Operator '::'	

Code	Old error	Quality	New error	New quality
PRAGMQA "In"	Error(s) found: before upper case identifier PRAGMQA at line 15, column 1 of file "testfile_mba. adl" Expecting lower case identifier ?lc? or "." or "=" or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PRAGMA" or "PURPOSE" or "R ELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" or ("MEANING" ...)* Try inserting symbol "CONCEPT" ========== before "MEANING" at line 16, column 1 of file "testfile_mba.adl" Expecting "::" Try deleting symbol "MEANING" at line 16, column 1 of file "testfile_mba.adl" ========== before string "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de ro l van werkgever vervult." at line 16, column 9 of file "testfile_mba.adl" Expecting "::" Try inserting symbol "::" ==========	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 15, column 1): unexpected Upper case identifier PRAGMQA expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
PURPOSE RELATION werkgever -{+Om de werkgever te kunnen bepalen gaan we ervan uit dat elke arbeidsrelatie precies n werkgever heeft.-}	Error(s) found: before "ENDPATTERN" at line 22, column 1 of file "testfile_mba.adl" Expecting symbol [or "HTML" or "IN" or "LATEX" or "MARKDOWN" or "REF" or "REST" or explanation {+-} Try inserting symbol explanation {+-}	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 22, column 1): unexpected Keyword "ENDPATTERN" expecting Symbol '[', Keyword "IN", Keyword "REST", Keyword "HTML", Keyword "LATEX", Keyword "MARKDOWN" or Keyword "REF"	
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = []	No warning, Should be a warning?		No warning, Should be a warning?	

Code	Old error	Quality	New error	New quality
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = [(0,0)]	Error(s) found: before "0" at line 17, column 5 of file "testfile_mba.adl" Expecting string "" Try deleting symbol "0" at line 17, column 5 of file "testfile_mba.adl" ========== before symbol , at line 17, column 6 of file "testfile_mba.adl" Expecting string "" Try inserting symbol string "" ========== before "0" at line 17, column 7 of file "testfile_mba.adl" Expecting string "" Try deleting symbol "0" at line 17, column 7 of file "testfile_mba.adl" ========== before symbol) at line 17, column 8 of file "testfile_mba.adl" Expecting string "" Try inserting symbol string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 17, column 5): unexpected Integer 0	

Code	Old error	Quality	New error	New quality
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = [("a","0")]	is this not an error??		is this not an error??	
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = [("a","0")	Error(s) found: before "PURPOSE" at line 18, column 1 of file "testfile_mba.adl" Expecting symbol] or (";" ...)* Try inserting symbol symbol]	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 18, column 1): unexpected Keyword "PURPOSE" expecting Operator ';' or Symbol ']'	
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." == [("a","0")]	Error(s) found: before "=" at line 17, column 2 of file "testfile_mba.adl" Expecting symbol [Try deleting symbol "=" at line 17, column 2 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 17, column 2): unexpected Operator '=' expecting Symbol '['	

Code	Old error	Quality	New error	New quality
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." == [("a","0");]	Expecting symbol [Try deleting symbol "=" at line 17, column 2 of file "testfile_mba.adl" ========== before symbol] at line 17, column 15 of file "testfile_mba.adl" Expecting symbol (Try deleting symbol symbol] at line 17, column 15 of file "testfile_mba.adl" ========== before "PURPOSE" at line 18, column 1 of file "testfile_mba.adl" Expecting symbol (Try inserting symbol symbol (========== before "PURPOSE" at line 18, column 1 of file "testfile_mba.adl" Expecting string "" Try inserting symbol string "" ========== before "PURPOSE" at line 18, column 1 of file "testfile_mba.adl" Expecting symbol , Try inserting symbol symbol , ========== before "PURPOSE" at line 18, column 1 of file "testfile_mba.adl" Expecting string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 17, column 2): unexpected Operator '=' expecting Symbol '['	

Code	Old error	Quality	New error	New quality
MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = [("a","0");("","")]	No warning, Should be a warning or error?		No warning, Should be a warning or error?	

Code	Old error	Quality	New error	New quality
werkgever :: Arbeidsrelatie -> Persoon BYPLUG [UNI, TRN] BYPLUG PRAGMA "In" MEANING IN ENGLISH "Elke arbeidsrelatie benoemt expliciet welke (rechts)persoon de rol van werkgever vervult." = [("a","0");("","")] DEFINE SRC "test"	Error(s) found: before upper case identifier DEFINE at line 18, column 1 of file "testfile_mba.a dl" Expecting lower case identifier ?lc? or "." or "CLASSIFY" or "CONCEPT" or "ENDPA TTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" Try inserting symbol "ENDPATTERN" ========== before upper case identifier DEFINE at line 18, column 1 of file "testfile_mba.a dl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol "RULE" ========== before "SRC" at line 18, column 8 of file "testfile_mba.adl" Expecting ":" Try inserting symbol ":"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 18, column 1): unexpected Upper case identifier DEFINE expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
RULE orderInAssortment :: orderOf |- orderedAt; sells MEANING "Products ordered at a vendor must be sold by that vendor" MESSAGE "A product was ordered at a vendor that does not sell it" VIOLATION (SRC orderedAt;vendorName, TXT " does not sell ", TGT productName)	Error(s) found: before "::" at line 223, column 24 of file "testfile_mba.adl" Expecting symbol [or lower case identifier ?lc? or "!" or "#" or "-" or "/" or "/\\" or ":" or ";" or "<>" or "=" or "CLASSIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" or "VIOLATION" or "\\" or "\\/" or "|-" or ("*" or "+" or " " ...)* or ("MEANING" ...)* or ("MESSAGE" ...)* Try deleting symbol "::" at line 223, column 24 of file "testfile_mba.adl" ========== before lower case identifier orderOf at line 223, column 27 of file "testfile_mb a.adl" Expecting ":" Try inserting symbol ":"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 223, column 24): unexpected Operator '::' expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "ROLE", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPROCESS"	

Code	Old error	Quality	New error	New quality
RULE 1onlyAcceptOwn : orderAccepted |- orderedAt	Error(s) found: before "1" at line 228, column 6 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or upper case identifier ?uc? o r "I" or "V" or string "" or atom ” or ("-" ...)* Try deleting symbol "1" at line 228, column 6 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 228, column 6): unexpected Integer 1 expecting Operator ’-’, Keyword "I", Keyword "V" or Symbol ’(’	
VIEW 1client: Client(clientName, TXT ", ", clientAddress, TXT " in ", clientCity)	Error(s) found: before "1" at line 48, column 6 of file "testfile_mba.adl" Expecting lower case identifier ?LC? or upper case identifier ?UC? or string "?S TR?" Try deleting symbol "1" at line 48, column 6 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 6): unexpected Integer 1	

Code	Old error	Quality	New error	New quality
VIEW Client: Client{Test}(clientName, TXT ", ", clientAddress, TXT " in ", clientCity)	Error(s) found: before symbol { at line 48, column 20 of file "testfile_mba.adl" Expecting symbol (Try inserting symbol symbol (========== before symbol { at line 48, column 20 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or upper case identifier ?UC? o r "I" or "PRIMHTML" or "TXT" or "V" or string "?STR?" or atom ’’ or ("-" ...)* Try inserting symbol lower case identifier ?LC? ========== before symbol (at line 48, column 26 of file "testfile_mba.adl" Expecting ":" Try deleting symbol symbol (at line 48, column 26 of file "testfile_mba.adl" ========== before lower case identifier clientName at line 48, column 27 of file "testfile_ mba.adl" Expecting ":" Try inserting symbol ":"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 25): unexpected Symbol ’}’ expecting Operator ’:’	

Code	Old error	Quality	New error	New quality
VIEW Client: Client, Client(clientName, TXT ", ", clientAddress, TXT " in ", clientCity)	Error(s) found: before symbol , at line 48, column 20 of file "testfile_mba.adl" Expecting symbol (Try deleting symbol symbol , at line 48, column 20 of file "testfile_mba.adl" ========== before upper case identifier Client at line 48, column 22 of file "testfile_mba. adl" Expecting symbol (Try deleting symbol upper case identifier Client at line 48, column 22 of file " testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 20): unexpected Symbol ',' expecting Keyword "DEFAULT", Symbol '{' or Symbol '('	

Code	Old error	Quality	New error	New quality
VIEW Client: Client(clientName, TXT ", "; clientAddress, TXT " in ", clientCity)	Error(s) found: before ";" at line 48, column 41 of file "testfile_mba.adl" Expecting symbol) or symbol , Try inserting symbol symbol , ========== before ";" at line 48, column 41 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or upper case identifier ?UC? o r "I" or "PRIMHTML" or "TXT" or "V" or string "?STR?" or atom '' or ("-" ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 41): unexpected Operator ';' expecting Symbol ',' or Symbol ')'	

Code	Old error	Quality	New error	New quality
VIEW Client: Client(clientName, TXdT ", ", clientAddress, TXT " in ", clientCity)	Error(s) found: before string ", " at line 48, column 38 of file "testfile_mba.adl" Expecting symbol { or ":" Try inserting symbol symbol { ========== before "TXT" at line 48, column 59 of file "testfile_mba.adl" Expecting (lower case identifier ?lc? or upper case identifier ?uc? or string "" ...)+ Try deleting symbol "TXT" at line 48, column 59 of file "testfile_mba.adl" ========== before symbol) at line 48, column 81 of file "testfile_mba.adl" Expecting symbol , or symbol } or (lower case identifier ?lc? or upper case iden tifier ?uc? or string "" ...)* Try deleting symbol symbol) at line 48, column 81 of file "testfile_mba.adl" ========== before lower case identifier vendorName at line 54, column 1 of file "testfile_m ba.adl" Expecting symbol }	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 38): unexpected String ", " expecting Symbol '{' or Operator ':'	

Code	Old error	Quality	New error	New quality
VIEWs Client: Client(Client:clientName, PRIMHTML ", ", clientAddress, TXT " in ", clientCity)	Error(s) found: before upper case identifier VIEWs at line 48, column 1 of file "testfile_mba.ad l" Expecting lower case identifier ?lc? or "." or "CLASSIFY" or "CONCEPT" or "ENDCO NTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try deleting symbol upper case identifier VIEWs at line 48, column 1 of file "te stfile_mba.adl" ========== before upper case identifier Client at line 48, column 7 of file "testfile_mba.a dl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol "IDENT"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 48, column 1): unexpected Upper case identifier VIEWs expecting Keyword "META", Keyword "PATTERN", Keyword "PROCESS", Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "INTERFACE", Keyword "SQLPLUG", Keyword "PHPPLUG", Keyword "PURPOSE", Keyword "POPULATION", Keyword "THEMES", Keyword "INCLUDE" or Keyword "ENDCONTEXT"	

Code	Old error	Quality	New error	New quality
INTERFACE Overview {client} (lient) : I[ONE]	Error(s) found: before symbol (at line 109, column 29 of file "testfile_mba.adl" Expecting ":" or "FOR" Try inserting symbol ":" ========== before ":" at line 109, column 37 of file "testfile_mba.adl" Expecting "!" or "#" or "-" or "/" or "/\\" or ";" or "<>" or "BOX" or "INTERFAC E" or "\\" or "\\/" or ("*" or "+" or " " ...)* Try inserting symbol "BOX" ========== before ":" at line 109, column 37 of file "testfile_mba.adl" Expecting symbol [Try inserting symbol symbol [========== before ":" at line 109, column 37 of file "testfile_mba.adl" Expecting lower case identifier ?LC? or upper case identifier ?UC? or string "?S TR?" Try inserting symbol lower case identifier ?LC? ========== before "INTERFACE" at line 120,	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 109, column 29): unexpected Symbol '(' expecting Keyword "FOR" or Operator ':'	

Code	Old error	Quality	New error	New quality
INTERFACE Client (clientName, clientAddress, clientCity, orderReceived) FORT Client, Customer, "test" : I[Client]	Error(s) found: before upper case identifier FORT at line 120, column 73 of file "testfile_mba.a dl" Expecting symbol { or ":" or "FOR" Try inserting symbol symbol { ========== before ":" at line 120, column 103 of file "testfile_mba.adl" Expecting symbol , or symbol } or (lower case identifier ?lc? or upper case iden tifier ?uc? or string "" ...)* Try inserting symbol symbol }	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 120, column 73): unexpected Upper case identifier FORT expecting Symbol '{', Keyword "FOR" or Operator ':'	
INTERFACE Client "test" (clientName, clientAddress, clientCity, orderReceived) FOR Client, Customer, "test" : I[Client]	Error(s) found: before string "test" at line 120, column 18 of file "testfile_mba.adl" Expecting symbol (or symbol { or ":" or "FOR" Try deleting symbol string "test" at line 120, column 18 of file "testfile_mba.a dl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 120, column 18): unexpected String "test" expecting Keyword "CLASS", Symbol '(', Symbol '{', Keyword "FOR" or Operator ':'	

Code	Old error	Quality	New error	New quality
INTERFACE Client (clientName, clientAddress, clientCity, orderReceived) FOR Client, Customer, "test" :: I[Client]	Error(s) found: before "::" at line 120, column 102 of file "testfile_mba.adl" Expecting symbol , or ":" Try deleting symbol "::" at line 120, column 102 of file "testfile_mba.adl" ========== before "I" at line 120, column 105 of file "testfile_mba.adl" Expecting ":" Try inserting symbol ":"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 120, column 102): unexpected Operator '::' expecting Symbol ',' or Operator ':'	

Code	Old error	Quality	New error	New quality
BOX <Test> ["Name" : clientName , "Street" : clientAddress	Error(s) found: before operator < at line 121, column 5 of file "testfile_mba.adl" Expecting symbol [Try deleting symbol operator < at line 121, column 5 of file "testfile_mba.adl" ========== before upper case identifier Test at line 121, column 6 of file "testfile_mba.ad l" Expecting symbol [Try deleting symbol upper case identifier Test at line 121, column 6 of file "te stfile_mba.adl" ========== before ">" at line 121, column 10 of file "testfile_mba.adl" Expecting symbol [Try deleting symbol ">" at line 121, column 10 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 121, column 5): unexpected Operator '<' expecting Symbol '<' or Symbol '['	

Code	Old error	Quality	New error	New quality
ROWS ["Name" : clientName , "Street" : clientAddress , "City" : clientCity	Error(s) found: before upper case identifier ROWS at line 121, column 1 of file "testfile_mba.ad l" Expecting "!" or "#" or "-" or "/" or "/\\" or ";" or "<>" or "BOX" or "INTERFAC E" or "\\" or "\\/" or ("*" or "+" or " " ...)* Try deleting symbol upper case identifier ROWS at line 121, column 1 of file "te stfile_mba.adl" ========== before symbol [at line 121, column 6 of file "testfile_mba.adl" Expecting "BOX" or "INTERFACE" Try inserting symbol "BOX"	Good	Geen issue!	

Code	Old error	Quality	New error	New quality
BOX [["All clients" : V[ONE*Client] , "All vendors" : V[ONE*Vendor] , "All products" : V[ONE*Product] , "All orders" : V[ONE*Order] BOX [product : orderOf;productName , client : orderedBy;clientName , vendor :orderedAt;vendorName]] INTERFACE test]	Error(s) found: before symbol [at line 110, column 6 of file "testfile_mba.adl" Expecting lower case identifier ?LC? or upper case identifier ?UC? or string "?S TR?" Try deleting symbol symbol [at line 110, column 6 of file "testfile_mba.adl" ========== before symbol] at line 119, column 18 of file "testfile_mba.adl" Expecting symbol (or symbol { or ":" or "FOR" Try deleting symbol symbol] at line 119, column 18 of file "testfile_mba.adl" ========== before "INTERFACE" at line 121, column 1 of file "testfile_mba.adl" Expecting ":" Try inserting symbol ":" ========== before "INTERFACE" at line 121, column 1 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 110, column 6): unexpected Symbol ’[’	

Code	Old error	Quality	New error	New quality
MEANING IN DUTCH RfEST {+ test -}	Error(s) found: before upper case identifier RfEST at line 225, column 18 of file "testfile_mba. adl" Expecting "HTML" or "LATEX" or "MARKDOWN" or "REST" or string "" or explanation {+-} Try deleting symbol upper case identifier RfEST at line 225, column 18 of file " testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 225, column 18): unexpected Upper case identifier RfEST expecting Keyword "REST", Keyword "HTML", Keyword "LATEX" or Keyword "MARKDOWN"	

Code	Old error	Quality	New error	New quality
MEANING REST IN DUTCH {+ test -}	Error(s) found: before "IN" at line 225, column 14 of file "testfile_mba.adl" Expecting string "" or explanation {+-} Try inserting symbol string "" ========== before "IN" at line 225, column 14 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT" or "KEY" or "MEANING" or "POPULATION" or "PURPOSE" or "RELATION" or " ROLE" or "RULE" or "SPEC" or "VIEW" or "VIOLATION" or ("MESSAGE" ...)* Try inserting symbol "MESSAGE"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 225, column 14): unexpected Keyword "IN"	

Code	Old error	Quality	New error	New quality
RULE OrderInAssortment : orderOf |- orderedAt; sells MESSAGE "A product was ordered at a vendor that does not sell it" MEANING IN DUTCH {+ test -} VIOLATION (SRC orderedAt;vendorName, TXT " does not sell ", TGT productName)	Error(s) found: before "MEANING" at line 226, column 1 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT" or "KEY" or "MESSAGE" or "POPULATION" or "PURPOSE" or "RELATION" or " ROLE" or "RULE" or "SPEC" or "VIEW" or "VIOLATION" Try inserting symbol "RULE" ========== before "MEANING" at line 226, column 1 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?lc? or upper case identifier ?uc? o r "I" or "V" or string "" or atom ” or ("-" ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 226, column 1): unexpected Keyword "MEANING" expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "ROLE", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPROCESS"	

Code	Old error	Quality	New error	New quality
VIOLATION (SRC orderedAt;vendorName+1, TXT " does not sell ", TGT productName)	Error(s) found: before "1" at line 227, column 37 of file "testfile_mba.adl" Expecting symbol) or "*" or "+" or "-" or "/" or "/\\" or ";" or "<>" or "\\" o r "\\/" or " " or (symbol , ...)* Try deleting symbol "1" at line 227, column 37 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 227, column 37): unexpected Integer 1 expecting Operator ' ', Operator '*', Operator '+', Operator ';', Operator '/', Operator '\', Operator '<>', Operator '-', Operator '/\', Operator '\/', Symbol ',' or Symbol ')'	

Code	Old error	Quality	New error	New quality
VIOLATION (SRC orderedAt;vendorName; TXT " does not sell ", TGT productName)	Error(s) found: before "TXT" at line 227, column 38 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try inserting symbol lower case identifier ?LC? ========== before "TXT" at line 227, column 38 of file "testfile_mba.adl" Expecting symbol) or symbol [or "-" or "/" or "/\\" or ";" or "<>" or "\\" or "\\/" or (symbol , ...)* or ("*" or "+" or " " ...)* Try inserting symbol symbol ,	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 227, column 38): unexpected Keyword "TXT" expecting Operator ’-’, Keyword "I", Keyword "V" or Symbol ’(’	

Code	Old error	Quality	New error	New quality
productPrice :: Product -> Price BYPLUG [UNI, PORP] BYPLUG	Error(s) found: before upper case identifier PORP at line 81, column 47 of file "testfile_mba.ad l" Expecting "ASY" or "INJ" or "IRF" or "PROP" or "RFX" or "SUR" or "SYM" or "TOT" or "TRN" or "UNI" Try deleting symbol upper case identifier PORP at line 81, column 47 of file "te stfile_mba.adl" ========== before symbol] at line 81, column 51 of file "testfile_mba.adl" Expecting "ASY" or "INJ" or "IRF" or "PROP" or "RFX" or "SUR" or "SYM" or "TOT" or "TRN" or "UNI" Try inserting symbol "UNI"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 81, column 47): unexpected Upper case identifier PORP expecting Keyword "UNI", Keyword "INJ", Keyword "SUR", Keyword "TOT", Keyword "SYM", Keyword "ASY", Keyword "TRN", Keyword "RFX", Keyword "IRF", Keyword "AUT" or Keyword "PROP"	
productPrice :: Product -> Price BYPLUG [UNI, PROP] BYPLUG PRAGMA	Error(s) found: before "=" at line 82, column 3 of file "testfile_mba.adl" Expecting (string "" ...)+ Try inserting symbol string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 82, column 3): unexpected Operator '='	

Code	Old error	Quality	New error	New quality
productPrice :: Product -> Price BYPLUG [UNI, PROP] BYPLUG PRAGMA "TEST" MEANING IN DUTCH {+ test -} = [("Product_1"; "10,00 euro") ; ("Product_2", "0,75 euro") ; ("Product_3", "6,95 euro") ; ("Product_4", "8,50 euro") ; ("Product_5", "4,50 euro")]	Expecting symbol , Try deleting symbol ";" at line 83, column 19 of file "testfile_mba.adl" ========== before string "10,00 euro" at line 83, column 21 of file "testfile_mba.adl" Expecting symbol , Try inserting symbol symbol ,	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 83, column 19): unexpected Operator ';' expecting Symbol ','	
productPrice :: Product -> Price BYPLUG [UNI, PROP] BYPLUG PRAGMA "TEST" MEANING IN DUTCH {+ test -} = [("Product_1", "10,00 euro") , ("Product_2", "0,75 euro") ; ("Product_3", "6,95 euro") ; ("Product_4", "8,50 euro") ; ("Product_5", "4,50 euro")]	Error(s) found: before symbol , at line 84, column 5 of file "testfile_mba.adl" Expecting symbol] or (";" ...)* Try deleting symbol symbol , at line 84, column 5 of file "testfile_mba.adl" ========== before symbol (at line 84, column 7 of file "testfile_mba.adl" Expecting symbol] or (";" ...)* Try inserting symbol ";"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 84, column 5): unexpected Symbol ',' expecting Operator ';' or Symbol ']'	

Code	Old error	Quality	New error	New quality
productName :: Product -> Name = ["Product_1" * "Inner tube" ; "Product_2" * "Outer tube"]	Error(s) found: before string "Product_1" at line 74, column 7 of file "testfile_mba.adl" Expecting symbol] or (symbol (...)* or (lower case identifier ?lc? or upper ca se identifier ?uc? or atom '' or decimal Integer 1 ...)* Try inserting symbol symbol] ========== before string "Product_1" at line 74, column 7 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "." or "CLASSIFY" or "CONCEPT" or "ENDCO NTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPLUG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol lower case identifier ?lc? ========== before string "Product_1" at line 74, column 7 of file "testfile_mba.adl" Expecting "::" Try inserting symbol "::"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 75, column 5): unexpected Operator ';' expecting Symbol ',' or Symbol ']'	

Code	Old error	Quality	New error	New quality
productName :: Product -> Name = ("Product_1" * "Inner tube" , "Product_2" * "Outer tube")	Error(s) found: before symbol (at line 74, column 5 of file "testfile_mba.adl" Expecting symbol [Try inserting symbol symbol [========== before "*" at line 74, column 19 of file "testfile_mba.adl" Expecting symbol , Try deleting symbol "*" at line 74, column 19 of file "testfile_mba.adl" ========== before string "Inner tube" at line 74, column 21 of file "testfile_mba.adl" Expecting symbol , Try deleting symbol string "Inner tube" at line 74, column 21 of file "testfile_ mba.adl" ========== before "*" at line 75, column 19 of file "testfile_mba.adl" Expecting symbol) Try deleting symbol "*" at line 75, column 19 of file "testfile_mba.adl" ========== before string "Outer tube" at	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 74, column 5): unexpected Symbol '(' expecting Symbol '[	

Code	Old error	Quality	New error	New quality
CONCEPT "PrestatieIndicator" BYPLUG "Een prestatie-indicator is een variable om prestaties van ondernemingen te analyseren." TYPE TESe	Error(s) found: before upper case identifier TESe at line 23, column 131 of file "testfile_mba.a dl" Expecting string "" Try deleting symbol upper case identifier TESe at line 23, column 131 of file "t estfile_mba.adl" ========== before "ENDPATTERN" at line 25, column 1 of file "testfile_mba.adl" Expecting string "" Try inserting symbol string ""	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 23, column 131): unexpected Upper case identifier TESe	
SPEC PrestatieIndicator ISA "Functie"""	Error(s) found: before string "" at line 26, column 38 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RUL E" or "SPEC" or "VIEW" Try deleting symbol string "" at line 26, column 38 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 26, column 38): unexpected String "" expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
CLASIFY PrestatieIndicator ISA "Functie"	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 26, column 38): unexpected String "" expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	Grind	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 26, column 1): unexpected Upper case identifier CLASIFY expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	New quality

Code	Old error	Quality	New error	New quality
{+Om de werkgever te kunnen bepalen gaan we ervan uit dat elke arbeidsrelatie precies n werkgever heeft.-}	Error(s) found: before "=" at line 86, column 3 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RUL E" or "SPEC" or "VIEW" Try inserting symbol "ENDPATTERN" ========== before "=" at line 86, column 3 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol "RULE" ========== before "=" at line 86, column 3 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?lc? or upper case identifier ?uc? o r "I" or "V" or string "" or atom ” or ("-" ...)* Try inserting symbol lower case	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 86, column 3): unexpected Operator ’=’ expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
CLASSIFY PrestatieIndicator ISA ("Functie" /\ "Functie")	Error(s) found: before symbol (at line 27, column 33 of file "testfile_mba.adl" Expecting upper case identifier ?uc? or string "" Try deleting symbol symbol (at line 27, column 33 of file "testfile_mba.adl" ========== before "/\\" at line 27, column 43 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RUL E" or "SPEC" or "VIEW" Try inserting symbol "ENDPATTERN" ========== before "/\\" at line 27, column 43 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDCONTEXT" or "IDENT" or "INCLUDE" or "INTERFACE" or "KEY" or "META" or "PATTERN" or "PHPPL UG" or "POPULATION" or "PROCESS" or "PURPOSE" or "RELATION" or "RULE" or "SPEC" or "SQLPLUG" or "THEMES" or "VIEW" Try inserting symbol "RULE"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 27, column 33): unexpected Symbol '('	

Code	Old error	Quality	New error	New quality
CLASSIFY PrestatieIndicator ISA "Functie" /\ "Functie"	Error(s) found: before "/\\" at line 27, column 43 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RUL E" or "SPEC" or "VIEW" Try deleting symbol "/\\" at line 27, column 43 of file "testfile_mba.adl" ========== before string "Functie" at line 27, column 46 of file "testfile_mba.adl" Expecting "ENDPATTERN" Try deleting symbol string "Functie" at line 27, column 46 of file "testfile_mba .adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 27, column 43): unexpected Operator '/\' expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
PATTERN CONTEXT DeliverySimple IN ENGLISH ENDCONTEXT ENDPATTERN	Error(s) found: before "CONTEXT" at line 29, column 1 of file "testfile_mba.adl" Expecting lower case identifier ?lc? or "CLASSIFY" or "CONCEPT" or "ENDPATTERN" or "IDENT" or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RUL E" or "SPEC" or "VIEW" Try inserting symbol "PURPOSE" ========== before "ENDCONTEXT" at line 31, column 1 of file "testfile_mba.adl" Expecting "HTML" or "LATEX" or "MARKDOWN" or "REF" or "REST" or explanation {+-} Try deleting symbol "ENDCONTEXT" at line 31, column 1 of file "testfile_mba.adl" ========== before "ENDPATTERN" at line 33, column 1 of file "testfile_mba.adl" Expecting explanation {+-} Try inserting symbol explanation {+-}	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 29, column 1): unexpected Keyword "CONTEXT" expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPATTERN"	

Code	Old error	Quality	New error	New quality
RULE allAccepted: I |- orderAccepted; orderAccepted <> bestelling!	Error(s) found: before "MEANING" at line 249, column 1 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 249, column 1): unexpected Keyword "MEANING" expecting Operator ’-’, Keyword "I", Keyword "V" or Symbol ’(’	

Code	Old error	Quality	New error	New quality
RULE allAccepted: I |- orderAccepted; orderAccepted <> bestelling!levering/pakje --	Error(s) found: before "/" at line 248, column 77 of file "testfile_mba.adl" Expecting symbol [or lower case identifier ?lc? or "!" or "-" or "/\\" or "CLAS SIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT" or "KEY" or "POPULATION" or "PURPO SE" or "RELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" or "VIOLATION" or "\\/ " or ("*" or "+" or " " ...)* or ("MEANING" ...)* or ("MESSAGE" ...)* Try inserting symbol "RULE" ========== before "/" at line 248, column 76 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?lc? or upper case identifier ?uc? o r "I" or "V" or string "" or atom '' or ("-" ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 248, column 76): unexpected Operator '/' expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "ROLE", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPROCESS"	

Code	Old error	Quality	New error	New quality
RULE allAccepted: (I |- orderAccepted; orderAccepted <> bestelling)	Error(s) found: before "|-" at line 248, column 22 of file "testfile_mba.adl" Expecting symbol) or symbol [or "!" or "#" or "-" or "/" or "/\\" or ";" or "< >" or "\\" or "\\/" or ("*" or "+" or " " ...)* Try inserting symbol symbol) ========== before symbol) at line 248, column 68 of file "testfile_mba.adl" Expecting symbol [or lower case identifier ?lc? or "!" or "#" or "-" or "/\\" o r ";" or "CLASSIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT" or "KEY" or "POPULAT ION" or "PURPOSE" or "RELATION" or "ROLE" or "RULE" or "SPEC" or "VIEW" or "VIOL ATION" or "\\/" or ("*" or "+" or " " ...)* or ("MEANING" ...)* or ("MESSAGE" .. .)* Try deleting symbol symbol) at line 248, column 68 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 248, column 22): unexpected Operator '|-' expecting Symbol '[', Operator ' ', Operator '*', Operator '+', Operator ';', Operator '!', Operator '#', Operator '/', Operator '\', Operator '<>', Operator '-', Operator '/\', Operator '\/' or Symbol ')'	

Code	Old error	Quality	New error	New quality
RULE allAccepted: I |- orderAccepted; orderAccepted- - == TOT	Error(s) found: before "MEANING" at line 249, column 1 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 249, column 1): unexpected Keyword "MEANING" expecting Operator ’-’, Keyword "I", Keyword "V" or Symbol ’(’	
RULE allAccepted: I ONE <> orderAccepted; orderAccepted - == TOT	Error(s) found: before "ONE" at line 248, column 21 of file "testfile_mba.adl" Expecting symbol [or lower case identifier ?lc? or "!" or "#" or "-" or "/" or "/\\" or ";" or "<>" or "=" or "CLASSIFY" or "CONCEPT" or "ENDPROCESS" or "IDENT " or "KEY" or "POPULATION" or "PURPOSE" or "RELATION" or "ROLE" or "RULE" or "SP EC" or "VIEW" or "VIOLATION" or "\\" or "\\/" or "|-" or ("*" or "+" or " " ...) * or ("MEANING" ...)* or ("MESSAGE" ...)* Try deleting symbol "ONE" at line 248, column 21 of file "testfile_mba.adl"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 248, column 21): unexpected Keyword "ONE" expecting Keyword "RULE", Keyword "CLASSIFY", Keyword "RELATION", Keyword "ROLE", Keyword "CONCEPT", Keyword "SPEC", Keyword "IDENT", Keyword "VIEW", Keyword "KEY", Keyword "PURPOSE", Keyword "POPULATION" or Keyword "ENDPROCESS"	

Code	Old error	Quality	New error	New quality
RULE allAccepted: ATOM <> orderAccepted; orderAccepted - == TOT	Error(s) found: before upper case identifier ATOM at line 248, column 19 of file "testfile_mba.a dl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try deleting symbol upper case identifier ATOM at line 248, column 19 of file "t estfile_mba.adl" ========== before "<>" at line 248, column 24 of file "testfile_mba.adl" Expecting symbol (or lower case identifier ?LC? or "I" or "V" or atom ’’ or ("- " ...)* Try inserting symbol lower case identifier ?LC?	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 248, column 19): unexpected Upper case identifier ATOM expecting Operator ’-’, Keyword "I", Keyword "V" or Symbol ’(’	

Code	Old error	Quality	New error	New quality
ROLE Client,Order MAINTAIN allReceived,allsent	Error(s) found: before upper case identifier MAINTAIN at line 261, column 19 of file "testfile_m ba.adl" Expecting symbol , or "EDITS" or "MAINTAINS" Try deleting symbol upper case identifier MAINTAIN at line 261, column 19 of fil e "testfile_mba.adl" ========== before lower case identifier allReceived at line 261, column 28 of file "testfil e_mba.adl" Expecting "EDITS" Try inserting symbol "EDITS"	Good	PE "ArchitectureAndDesign/Syntax/testfile_mba.adl" (line 261, column 19): unexpected Upper case identifier MAINTAIN expecting Symbol ',', Keyword "MAINTAINS" or Keyword "EDITS"	

Table 3: Complete error list with their respective qualities

References