Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 129 (2024) 49–54

www.elsevier.com/locate/procedia

18th CIRP Conference on Computer Aided Tolerancing (CAT2024)

GPS&V Textual Language

Jean-François Maurel

ITG Formation 34, rue Laffitte 75009 PARIS FRANCE

* Corresponding author. Tel.: +33 974 77 00 90. *E-mail address:* jf.maurel@velama.fr

Abstract

The ISO Technical Committee 213 (TC 213) develops standards about geometrical product specifications and verification (GPS&V). Those standards define both the meaning (semantic) and the writing (syntax) of specification indications used in Technical Product Documentation to control the size, geometry and surface texture deviation allowance on a mechanical part. The specification indications are defined as graphical symbols whose dimension and proportion are described in different ISO standards. The arrangement of those symbols is standardized so that reliable if not univocal information can be conveyed from the writer of the specification indication to the readers of the specification indication. This set of standardized rules constitutes a Graphical Language, primarily developed for humans widely used on two dimensional drawings or in three dimensional models. However, since the specifications are becoming more and more complex it is foreseen that the machines will be more involved in the communication process of the geometrical tolerancing information in the future. This paper describes in some details a context-free grammar that has been written to test the feasibility of a machine-readable Textual Language development.The grammar is based on the Universal Coded Character set (UCS). It shall be able to convey the same information as the current Graphical Language defined in ISO standards. The paper also reports on some testing tools that have been developed and made publicly available through a web interface. While this paper is only concerned with the syntax of the specification indications, some proposals are presented for an extended use of the grammar to check also semantic rules.

Keywords: geometrical tolerancing ; context-free grammar ; ISO standard

1. Introduction

Current ISO standards define the symbols needed to write ISO specification indications on drawings or annotation in numerical models as graphical symbols. The symbols are defined in annexes such as ISO 1101:2017 Annex F and by reference to standards such as ISO 81714 or ISO 3098. The shape and dimension of graphical symbols are typically defined through grids as shown in Figure 1. The

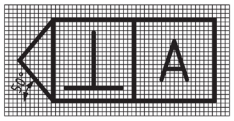


Fig. 1: Graphical symbol definition example.

graphical symbol definitions together with the rules for the layout of those symbols and the meaning associated to those symbols set by ISO standards constitute a Graphical Language. Historically, this Graphical Language has been developed to exchange information between humans. Communication was originally achieved through paper documents however the current practise is to use digitalized documents issued or displayed with a Computer Aided Design system ('CAD system'). CAD systems are able to deal efficiently with graphical information either in the form of rasterized graphics (where the graphic is modelled as a set of pixels) or of vectorized graphics (where the graphic is modelled as a set of geometric primitives). However, while humans can easily differentiate graphical symbols that contain some deviation from the standard shape, a robust recognition of a set of graphical symbols with deviations is not so easy for a machine. This is particularly true when the source of information is unknown so that we have to rely on the graphical information only. This can lead to some difficulty when developing automatic tools

10.1016/j.procir.2024.10.010

across different systems to assist users in achieving quickly and reliably tasks such as:

- searching for a full set of graphical symbols inside a file
- searching for a specific symbol inside a set of graphical symbols
- linking a standardized meaning (semantic) to a set of symbols

Other drawbacks are encountered during the development or use of ISO standards as for example:

- accurate definition of a graphic symbol through grids is cumbersome and not accurate
- consistent display of graphical symbols in different standards is problematic
- search inside ISO standards is difficult when the information is embedded in graphics
- automatic testing of a new feature or new symbol added in a standard can be difficult or even impossible

We consider, in this paper the possibility of using text characters to convey the same information as the Graphical Language. The usage of textual characters for the specification indications, could greatly improve the machine-readability while keeping a certain level of human-readability. The ISO/IEC 10646 International Standard [1], publicly available from ISO web site, defines a Universal Coded character Set (UCS) also known as Unicode [2] character set. This set encompasses much if not all of the alphabets, a large number of characters used in the world for scripting and also a lot of other technical and specific symbols. The representation of specification indications with textual characters has already drawn some attention as described in [8]. In contrast to graphical symbols defined by a possibly large set of pixels or geometric primitives, the coded characters are defined with a single number. Therefore, the coded characters are easily identified by a machine even inside a large file. On the downside the appearance of the coded characters is not standardized as it depends on the actual font chosen for the character display. Therefore, the recognition by a human reader of a coded character without ambiguity on a display device is submitted to the definition of a standard font. The rendering of graphic symbols and graphics in general is well managed by machines on different display devices. Several standardized formats exist both for raster graphics and vectorized graphics. In the case of coded characters, we have to take into account how the coded characters are actually stored by the software used and eventually by the operating system installed on the user machine. ISO/IEC 10646 specifies different encoding schemes for textual characters. UTF-8 is probably the most widely used encoding scheme for text and is available on all commonly used operating systems. Nevertheless, while UTF-8 is quite ubiquitous in current software, it is not always the default encoding scheme of operating systems. Therefore, the software or the user has to take care of encoding translations if needed, for example between CP-1252 and UTF-8. The encoding translation is a trivial task for professional text editors but has to be checked when using other types of text editors. The purpose of the current document is to explore the feasibility of developing a grammar for a Textual Language. It is focused on the development of an invisible XML grammar (see Appendix B). The grammar syntax rules are expressed with a set of coded characters out of UCS. Semantic rules are not considered.

2. Overall technical description

A fully operational grammar (see Appendix A) has been developed using a new grammar syntax called invisible XML. Grammars are of common use in industry for the development of programming language compilers. The purpose here, is to apply those well-known techniques in the field of geometrical tolerancing. The Textual Language grammar presented is used to parse a sequence of characters in order to check their conformity against the grammar rules. This task is achieved with a specific parser that is able to read Invisible XML syntax grammars. The parser input data consist of a text fragment made of a sequence of characters from UCS called a textual indication, and of the Textual Language grammar in invisible XML syntax. The text encoding scheme used for the whole process is UTF-8. The input data can be prepared directly by the user with a text editor equipped with Unicode fonts and able to store its content as UTF-8. For the sake of testing and also illustrating the parsing process, a user interface (UI) called GPS&V TIP has been developed to assist in preparing and parsing data. This UI is freely available. Once the input data called a textual indication is available, the next step is to feed an invisible XML parser with this data. This is achieved simply by clicking ‘submit’ in GPS&V TIP. Several open-source parsers have been already developed for the invisible XML grammars. The output of the invisible XML parser is either an error message or, an eXtensible Markup Language (XML) fragment when the parsing is successful. The error message is either due to an ambiguity in the grammar or to a syntax error contained in the textual indication. In general, an input data can reveal some ambiguity contained in the grammar rule definitions (See [11]). The existence of such an ambiguity means that there is more than one way for the grammar rules to match a specific input data. However, a fully tested invisible XML grammar can be assumed to be free of any ambiguities.

The XML element names and the hierarchy of the elements in the XML output resulting from the parsing of the textual indication are derived from the Textual Language grammar. Therefore, the grammar developer has full control over the content of the XML output. This is considered

as a great feature when dealing with human readability and also in the perspective of standardizing.

Once the syntax of a particular textual indication has been checked then an XML fragment is automatically obtained (see subsection 4.2). This XML is nothing more than another representation of the textual indication that was originally input. However, this XML structure can be conveniently used in a further step to check semantic rules. This can be achieved for example by feeding a subsequent transformer software with the XML output obtained to perform semantic checking (see subsection 4.3). Post-processing of the XML is used in GPS&V TIP to obtain a graphical-like representation of the data.

3. Textual Language grammar design

3.1. Motivation

The purpose of designing a grammar for a Textual Language is to check whether an ISO specification indication can be translated into a textual representation. The grammar shall use only embedded syntax rules and UCS. The starting point for the design of the Textual Language grammar described here after is the description of a geometrical specification indication stated in ISO 1101:2017 paragraph 8. This description of the layout of an ISO geometrical specification indication is taken as a basis to derive syntax rules for the Textual Language. The description applies to geometrical specification but also includes an optional size specification indication or a surface texture specification indication. Therefore, a general pattern for a Textual Language indication has been derived from this description.

3.2. General pattern

Figure 2 presents the correspondence between the information included in an ISO geometrical specification indication and the main components of the Textual Language grammar.

The first line of the grammar defines the main rule, whose name is `indication` and gives the general pattern for a textual indication, namely:

```
1  indication : connection+ , (profile ; surface ;
       size ; folr) .
```

Listing 1: indication grammar rule

According to this rule, an `indication` is (according to the colon symbol) a sequence of at least one (according to the + symbol) `connection` and (according to the comma symbol) a profile surface texture indication `profile` or (according to the semi colon symbol) an areal surface texture indication `surface` or (according to the semi colon symbol) a size specification indication (`size`) or (according to the semi colon symbol) a form, orientation, location or run-out specification indication `folr` in short.

Each rule name (`indication`, `connection`, `profile`, `surface`, `size`, `folr`) contained in the definition of an indication is further defined by a rule in the grammar. Then the rule is recursively expanded with subsequent rule definitions down to the occurrence of a terminal. A terminal does not contain any further rule definition but only literal strings. We describe in more details some of the main grammar rules. The usage of the grammar is described in section 4.

3.3. Main rules

3.3.1. connection

The rule `connection` is designed to convey information about the nominal model (`nominal`) and the type of connectors (`connector`) between the nominal model and the specification indication. A `connection` is delimited with brackets (`open` and `close`). Therefore, the grammar rule is as follows:

```
1  connection : open , nominal , connector , close .
```

Listing 2: connection grammar rule

The current version of the Textual Language grammar contains only placeholders for the rules describing the nominal model. It is expected that in some scenarios the information about the nominal will be readily available from the context so that the nominal model rule will not be helpful. Otherwise, additional rules will have to be written in order to be able to define the toleranced feature.

3.3.2. Form, orientation, location or run-out

The form, orientation, location or run-out textual indication (`folr`) is made of an optional upper area (`upper_folr_area`), a main specification indication line (`main_folr_line`) and zero or more stacked specification indication lines (`stacked_folr_line`). The grammar rule is as follows:

```
1  folr : upper_folr_area? , main_folr_line ,
       stacked_folr_line* .
```

Listing 3: folr grammar rule

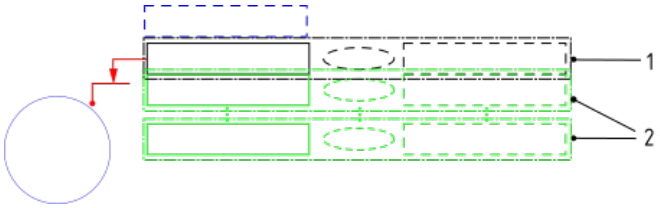
A lower area has not been added in the grammar as it would contain the same information as an upper area. Furthermore, upper and lower area are mutually exclusive in the geometrical specification indications so that only the upper area has been retained in the Textual Language.

3.3.3. Size

The textual `size` indication is aligned on the textual `folr` specification indication therefore the grammar rules is as follows:

```
1  size : upper_size_area? , main_size_line ,
       stacked_size_line* .
```

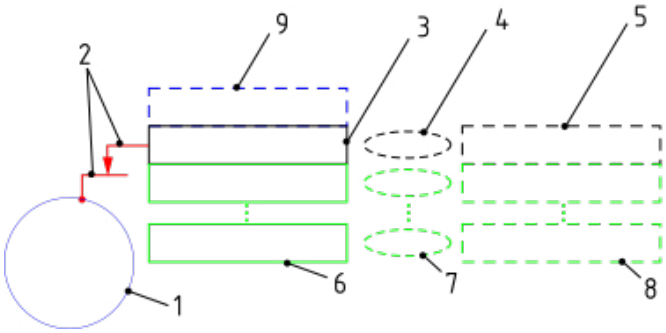
Listing 4: size grammar rule



Keys:

1	Main line (main_folr_line)
2	Stacked lines (stacked_folr_line)

(a) Specificaton indications (lines) stacking



Keys:

1	Nominal model (nominal)
2	Connection between nominal model and specification indication (connection)
3	Main line frame (frame)
4	Main line planes and feature indication area (plane_feature_area)
5	Main line inline area (inline)
6	Stacked line frame (frame)
7	Stacked line planes and feature indication area (planes_feature_area)
8	Stacked line inline area (inline)
9	Upper indication area (upper_folr_area)

(b) Main components of a specification indication

Fig. 2: Textual Language grammar names and ISO geometrical specification components

As a side note, it appears that the size rule is a child of the indication grammar rule and also a child of `upper_folr_area`. This design option has been selected in order to mimic the current definition in ISO 1101. However, it makes the grammar rules over-complicated even if the tests already developed passed with the current version of the grammar.

3.3.4. Surface texture

The textual indications for surface texture (areal or profile) are included in the grammar. The rule for a profile surface texture indication is:

```
1  profile        : upper_profile_line? ,
       first_profile_line , second_profile_line? ,
       profile_direction_line? .
```

Listing 5: profile grammar rule

The areal surface texture indication grammar rule is not detailed in the current version of the grammar and is currently simply defined by a placeholder as:

```
1  surface        : "s" .
```

Listing 6: surface grammar rule

4. Textual Language grammar usage

The current version of the Textual Language grammar has been developed by considering the following ISO standards:

- ISO 1101:2017
- ISO 5459:2011
- ISO 2692:2021
- ISO 5458:2018
- ISO 3040:2016
- ISO 14405-1: 2016
- ISO 21920-1:2021

The grammar has been tested against the full set of examples from:

- ISO 1101:2017
- ISO 5459:2011
- ISO 2692:2021
- ISO 5458:2018
- ISO 3040:2016

The ability to automatically run tests for a specific grammar is considered as an important feature for the design of a new Textual Language and in the context of standards development.

The Textual Language grammar can be used to check the syntax conformance of an input data by processing the following steps:

- Input data preparation
- Input data parsing
- Output analysis
- Post-processing of the XML output (Optional)

4.1. Input data preparation

Input data shall be a sequence of UTF-8 characters that are expected to be conformant to the Textual Language grammar. This sequence could be directly typed by the user in a text editor that is UCS capable. However, user-interface can easily be written to assist the user in this task. [3] shows an example of an Hyper-Text Markup Language (HTML) page empowered with some JavaScript code. We consider a straightness ISO specification indication in order to illustrate the usage of the grammar. We show how we can translate this graphical indication into a textual indication and then parse it. The graphical indication is shown in Figure 3:

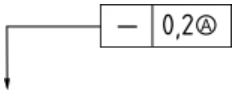


Fig. 3: Straightness specification graphical indication

This is translated in Textual Language as:

```
{line2d_12►0}{{{—}{0,2Ⓐ}}}
```

Listing 7: Textual indication corresponding to Figure 3

The grammar parser can now be fed with the input data from Listing 7 to check the syntax.

4.2. Output analysis

After parsing the character sequence in Listing 7 against the Textual Language grammar with GPS&V TIP, we obtain the following XML fragment as output:

```
<?xml version="1.0" encoding="UTF-8"?>
<indication>
   <connection>
      <nominal>
         <dim2>line2d_12</dim2>
      </nominal>
      <connector>
         <leader_line>►0</leader_line>
      </connector>
   </connection>
   <folr>
      <main_folr_line>
         <frame>
            <symbol_cell>—</symbol_cell>
            <tol_cell>0,2Ⓐ</tol_cell>
         </frame>
      </main_folr_line>
   </folr>
</indication>
```

Listing 8: Listing 7 parsing result

Obtaining this XML fragment as a result means that the input data are conformant to the grammar syntax. This XML also sheds some light on the tree structure of the textual indication and its main components.

4.3. Post-processing

XML fragment can be conveniently transformed with some eXtensible Stylesheet Language Transformations (XSLT). XSLT is very powerful and widely used to process XML files. An example of post-processing is shown in GPS&V TIP. XML output is further processed with XSLT to obtain an HTML fragment used by the browser to display a graphic-like representation of the input data. More standardized graphical indication could be obtained with a more sophisticated XSLT.

The power of XSLT can be used to further check the semantic of the textual indication. This stage is not described here as it is a work in progress which is not mature enough for now. However, some successful testing has been achieved. The idea is to extract semantic rules from ISO standards and translate them as XSLT templates. A remaining question is whether the information and particularly the geometrical information from the nominal model should be entirely embedded in the textual indication. This information could probably, be more conveniently grabbed from the context, for example a STEP ('STandard for the Exchange of Product model data') file [9].

4.4. Stacked specification indications example

We present here after stacked Graphical Language specification indications and the corresponding Textual Language indication. The graphical indication is presented in Figure 4.

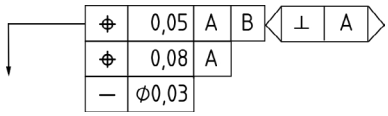


Fig. 4: Stacked graphical indications

The textual indication corresponding to the graphical indication shown in Figure 4 is presented in Listing 9:

The XML result obtained when feeding GPS&V TIP with the data from Listing 9, is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<indication>
 <connection>
  <nominal>
   <dim2>line2d_10</dim2>
  </nominal>
  <connector>
   <leader_line>►270</leader_line>
  </connector>
 </connection>
 <folr>
  <main_folr_line>
   <frame>
    <symbol_cell>⌖</symbol_cell>
    <tol_cell>0,05</tol_cell>
    <datum_section>
     <datum_cell>A</datum_cell>
```

```
{line2d_10►270}{{{⏚}{0,05}{A}{B}}{◁▷⊥A}}{{{⏚}{0,08}{A}}}{{{—}{⌀0,03}}}
```

Listing 9: Stacked specifications textual indication (see Figure 4)

```
      <datum_cell>B</datum_cell>
     </datum_section>
    </frame>
    <planes_feature_area>
     <orientation_plane>⊥A</orientation_plane>
    </planes_feature_area>
   </main_folr_line>
   <stacked_folr_line>
    <frame>
     <symbol_cell>⏚</symbol_cell>
     <tol_cell>0,08</tol_cell>
     <datum_section>
      <datum_cell>A</datum_cell>
     </datum_section>
    </frame>
   </stacked_folr_line>
   <stacked_folr_line>
    <frame>
     <symbol_cell>—</symbol_cell>
     <tol_cell>⌀0,03</tol_cell>
    </frame>
   </stacked_folr_line>
  </folr>
</indication>
```

Listing 10: Listing 9 parsing result

Acknowledgements and conclusion

A user-interface named GPS&V TIP has been developed. jωiXML Processor is used to parse the iXML grammar defining the Textual Language. Then, an XSLT transformation is launched through the use of SAXON JS in the browser. This process is inspired from Saxon-JS 2.5 XSLT 3 Fiddle from Martin Honnen

Appendix A. Textual Language grammar

The full Textual Language grammar [5] in Invisible XML syntax is publicly available and is published under General Public License (GNU).

Appendix B. Invisible XML

B.1. Specification

Invisible XML is a context free grammar language that is specified in [4]. A World Wide Web Consortium (W3C) community group [10] is currently working on Invisible XML specification development.

B.2. Presentation

A comprehensive description of Invisible XML grammar syntax implies the reading of the whole invisible XML specification document. However, we present here a basic introduction taken from [6]. [7] also contains useful information and tutorials.

An Invisible XML grammar is a list of rules. Each rule is made of a name followed by a colon and ended with a full stop. The first name is the grammar name and then each name in the rule has to be defined by a subsequent rule down to some terminals. Terminals are characters delimited by single quotes or double quotes and don't contain any further rule. The following characters, when not quoted have special meaning for the grammar:

- plus sign (+) means one or more occurrence of the preceding element
- question mark (?) means zero or one occurrence of the preceding element
- star (*) means zero or more occurrence of the preceding element
- comma (,) is a logical operator and
- semi-colon (;) is a logical operator or
- dash (-) before a name means that we want to exclude this name from the element names list in the final XML output

Bracketed text such as {this is a comment} is used to add comments in the grammar. Ranges of characters can be defined with square brackets like ["A"-"Z"] for capital letters. Encoded characters can be defined with their hexadecimal code number like 24BA for Ⓔ.

B.3. Parsers

Parsers for Invisible XML grammars are available ([7]).

References

[1] ISO 10646 Information technology - Universal coded character set
[2] Unicode Web Site
[3] GPS&V TIP (Textual Indication Parser)
[4] Invisible XML specification
[5] Textual Language grammar
[6] On the specification of Invisible XML, Steven Pemberton 2019
[7] Invisible XML web site
[8] Recommended Practices for PMI Unicode String Specifications J Boy, P Roché, R Astheimer, R Lipman
[9] ISO 10303 Industrial automation systems and integration Product data representation and exchange
[10] Invisible Markup Community Group
[11] Ambiguity in iXML: and how to control it. N Tovey-Walsh. 2023.