# 

# Data Format Description Language (DFDL) v1.0

**Experience Document 1**

**Errata for DFDL v1.0 Specification GFD.207**

Status of This Document

Grid Working Document (GWD)

Copyright Notice

Copyright © Open Grid Forum 2014 - 2019. Some Rights Reserved. Distribution is unlimited.

Abstract

This document provides experience information to the OGF community on the Data Format Description Language (DFDL) 1.0 specification (GFD-P-R.207).

It lists and describes the non-editorial errata identified in the DFDL 1.0 specification since its publication in September 2014.

**Contents**

[1. Introduction 2](#_Toc21705938)

[2. Minor Technical Fixes 3](#_Toc21705939)

[3. Security Considerations 16](#_Toc21705941)

[4. Contributors 17](#_Toc21705942)

[5. Intellectual Property Statement 18](#_Toc21705943)

[6. Disclaimer 19](#_Toc21705944)

[7. Full Copyright Notice 20](#_Toc21705945)

[8. References 21](#_Toc21705946)

# Introduction

This document has been created to list issues encountered by implementers of the DFDL 1.0 specification GFD.207 [DFDL], and users of implementations of the DFDL 1.0 specification. Specifically, it records all those issues requiring a non-editorial change to the DFDL 1.0 specification, in the form of errata.

The OGF GFD process [GFD] recognises three different kinds of error that may be found in OGF specifications:

*Editorial fixes*. Updates to a document which are not widely announced or publicized.

This category might include headers/footers, spelling, formatting, or simple wording

changes for clarity.

*Minor technical fixes.* Updates to a document which are not simply editorial. For example,

an update to an XML schema or addition to a protocol, to bring the document into

agreement with current practice.

*Major technical fixes.* Such fixes will often require additional technical review and result

in an updated or replaced document.

The following sections of this document list the errata that fall into the last two categories.

All errata are tracked by Redmine Issue trackers [ISSUES].

# Minor Technical Fixes

The following minor technical fixes have been identified. Note that they are numbered **5.x** to follow-on from errata numbering in [DFDLX1].

**5.1**. *Section 13.2.1.* [*https://redmine.ogf.org/issues/230*](https://redmine.ogf.org/issues/230)

Consider a dfdl:escapeScheme annotation with the following properties:

* dfdl:escapeBlockStart="start"
* dfdl:escapeBlockEnd="end"
* dfdl:escapeEscapeCharacter="#"

If this is used to serialize a DFDL Infoset element of type xs:string with value “A hash is a #”, then the value is wrapped with the dfdl:escapeBlockStart and dfdl:escapeBlockEnd, giving simple content "startA hash is a #end".  If this data was parsed, the "#end" will be treated as an escaped escape block end and the parse will fail, reporting that there is no escape block end in the data.   
  
In this scenario, the data is not compliant with the escape scheme, and the DFDL serializer must issue a processing error.

***5.2****. Section 23.* [*https://redmine.ogf.org/issues/231*](https://redmine.ogf.org/issues/231)

XPath 2.0 specification [XPATH2] allows implementation-dependent evaluation of expressions thereby allowing either lazy (sequential) evaluation or full (parallel) evaluation of expressions with OR and AND clauses. This flexibility is not desirable in DFDL 1.0 implementations so the specification is changed to prescribe lazy (sequential) evaluation left-to-right.

**5.3**. *Section 13.2.1.* [*https://redmine.ogf.org/issues/237*](https://redmine.ogf.org/issues/237)

Examples of formats exist where an escape character is active when in front of an in-scope delimiter, but not when in front of another character. Currently DFDL 1.0 will remove all instances of escape characters when parsing regardless of what follows. In order to handle these formats, a new property is added to dfdl:escapeScheme.

|  |  |
| --- | --- |
| **Property Name** | **Description** |
| escapeCharacterPolicy | Enum  Valid values are ‘all’, ‘delimiters’.  Controls when escape characters are removed during parsing, and output during unparsing, when dfdl:escapeKind is 'escapeCharacter'.  When 'all':  During unparsing the following are escaped as described in dfdl:escapeKind when they are in the data.   * Any in-scope terminating delimiter by escaping its first character. * dfdl:escapeCharacter (escaped by dfdl:escapeEscapeCharacter) * any dfdl:extraEscapedCharacters   During parsing, occurrences of dfdl:escapeCharacter and dfdl:escapeEscapeCharacter are interpreted and removed from the data as described in dfdl:escapeKind.  When 'delimiters':  During unparsing the following are escaped as described in dfdl:escapeKind when they are in the data.   * Any in-scope terminating delimiter by escaping its first character. * dfdl:escapeCharacter (escaped by dfdl:escapeEscapeCharacter)   During parsing, occurrences of dfdl:escapeCharacter and dfdl:escapeEscapeCharacter are interpreted and removed from the data as described in dfdl:escapeKind, except that dfdl:escapeCharacter is only removed when it immediately precedes an in-scope terminating delimiter.  Annotation: dfdl:escapeScheme |

Consequential updates to description of dfdl:escapeKind :

“When 'escapeCharacter': On unparsing a single character of the data is escaped by adding a dfdl:escapeCharacter or dfdl:escapeEscapeCharacter immediately before it. The characters to escape are determined by property dfdl:escapeCharacterPolicy.

On parsing any in-scope terminating delimiter encountered in the data is not interpreted as such when it is immediately preceded by the dfdl:escapeCharacter (when not itself preceded by the dfdl:escapeEscapeCharacter). Occurrences of the dfdl:escapeCharacter and dfdl:escapeEscapeCharacter are removed from the data as determined by property dfdl:escapeCharacterPolicy unless the dfdl:escapeCharacter is preceded by the dfdl:escapeEscapeCharacter, or the dfdl:escapeEscapeCharacter does not precede the dfdl:escapeCharacter, respectively.”

**5.4**. *Section 23.5.2.5*. [*https://redmine.ogf.org/issues/239*](https://redmine.ogf.org/issues/239)

The description of fn:exactly-one() does not match the XPath 2.0 specification [XPATH2]. It should state : “Returns the input sequence if it contains exactly one item. Raises an error otherwise.”

**5.5**. *Section 23*. [*https://redmine.ogf.org/issues/240*](https://redmine.ogf.org/issues/240)

XPath 2.0 specification [XPATH2] defines its functions to be in namespace <http://www.w3.org/2005/xpath-functions>. The DFDL specification must bind namespace prefix “fn:” to this namespace.

**5.6**. *Section 5.1*. [*https://redmine.ogf.org/issues/241*](https://redmine.ogf.org/issues/241)

The list of XSDL 1.0 constructs supported by DFDL 1.0 includes the attributes “elementFormDefault” and “form”. These should be explicitly listed.

**5.7**. *Section 13.6.* [*https://redmine.ogf.org/issues/238*](https://redmine.ogf.org/issues/238)

The paragraph in the description of the textNumberPadCharacter property that describes what happens when the pad character is ‘0’ does not cover edge cases such as a sign being present. It is revised as follows:

"When parsing, if the pad character is '0' and dfdl:textTrimKind is 'padChar' then the SimpleContent region is trimmed of the '0' characters as defined by the trimming rules. If at least one '0' character is removed and the trimmed text causes a processing error when parsed, a single '0' character is re-instated and the text is parsed again. This is to handle the case when '0' characters are trimmed away leaving no digits. This rule also applies when the pad character is a DFDL character entity equivalent to '0'. This rule does not apply when the pad character is any other character nor when a pad byte is specified."

**5.8.** *Sections 14.2, 16.* [*https://redmine.ogf.org/issues/243*](https://redmine.ogf.org/issues/243)

Following changes needed to the specification to clarify the behaviour of separator suppression. If it is applicable, the separatorSuppressionPolicy should apply to the whole sequence regardless of the occursCountKind of child elements.

*14.2* - updates to paragraphs that describe positional and non-positional sequences

***“Positional sequence*** - Each occurrence in the sequence can be identified by its position in the data. Typically the components of such a sequence do not have an initiator. In some such sequences, the separators for optional zero-length occurrences may or must be omitted when at the end of the group. In DFDL, a sequence is considered positional if it contains only required elements and/or optional and array elements that have dfdl:occursCountKind 'implicit', 'fixed' or 'expression', and it has dfdl:separatorSuppressionPolicy 'never', 'trailingEmptyStrict'  or 'trailingEmpty'.”

***“Non-positional sequence*** - Occurrences in the sequence cannot be identified by their position in the data alone. Often the components of such a sequence have an initiator. Such sequences sometimes allow the separator to be omitted for optional zero-length occurrences anywhere in the sequence. Speculative parsing might need to be employed by to identify each occurrence. In DFDL, a sequence is non-positional if it contains any optional or array elements that have dfdl:occursCountKind 'parsed' or 'stopValue', and/or it has dfdl:separatorSuppressionPolicy 'anyEmpty'.”

*14.2.2* - updates to the last sentences of the 'When dfdl:occursCountKind is ...' paragraphs to match the table and notes.

“When an element is required and is not an array then one occurrence is always expected along with its separator. The dfdl:separatorSuppressionPolicy of the sequence has no effect (nothing is eligible for suppression). Otherwise the behaviour is dependent on dfdl:occursCountKind.  
When dfdl:occursCountKind is 'fixed' then XSDL minOccurs must equal maxOccurs and that many occurrences are always expected along with their separators. The dfdl:separatorSuppressionPolicy of the sequence has no effect (nothing is eligible for suppression).  
When dfdl:occursCountKind is 'expression' the number of occurrences is given by dfdl:occursCount and exactly that many occurrences are always expected along with their separators. The dfdl:separatorSuppressionPolicy of the sequence has no effect (nothing is eligible for suppression).  
When dfdl:occursCountKind is 'parsed' any number of occurrences and their separators are expected. The dfdl:separatorSuppressionPolicy of the sequence must be 'anyEmpty' and it is a schema definition error otherwise.  
When dfdl:occursCountKind is 'stopValue', any number of occurrences and their separators are expected followed by the stop value and its separator. The dfdl:separatorSuppressionPolicy of the sequence has no effect.   
When dfdl:occursCountKind is 'implicit', between XSDL minOccurs and maxOccurs (inclusive) occurrences and their separators are expected, according to the dfdl:separatorSuppressionPolicy of the sequence.”

*14.2.3* - updates to the last sentences of the 'When dfdlk:occursCountKind is ...' paragraphs to match the table and notes.

When an element is required and is not an array then one occurrence is always output along with its separator. The dfdl:separatorSuppressionPolicy of the sequence has no effect (nothing is eligible for suppression).  
Otherwise the behaviour is dependent on dfdl:occursCountKind.  
When dfdl:occursCountKind is 'fixed' or 'expression' the occurrences in the augmented Infoset are always output along with their separators. The dfdl:separatorSuppressionPolicy of the sequence has no effect (nothing is eligible for suppression).  
When dfdl:occursCountKind is 'parsed' non zero-length occurrences in the augmented Infoset are output along with their separators. The dfdl:separatorSuppressionPolicy of the sequence must be 'anyEmpty' and it is a schema definition error otherwise.  
When dfdl:occursCountKind is 'stopValue' the occurrences in the augmented Infoset are output along with their separators followed by the stop value and its separator, according to the dfdl:separatorSuppressionPolicy of the sequence.  
When dfdl:occursCountKind is 'implicit' the occurrences in the augmented Infoset are output along with their separators, according to the dfdl:separatorSuppressionPolicy of the sequence.

*16* - update to occursStopValue property description. The property is a list of logical values, so need to add: "The dfdl:stopValue property must not be empty string."

**5.9**. *Section 16.1.* [*https://redmine.ogf.org/issues/244*](https://redmine.ogf.org/issues/244)

New clauses added to sections 16.1.1 through 16.1.6 where needed to describe the behaviour when maxOccurs or occursCount is zero.

*16.1.1* (fixed):  
Parsing: When maxOccurs is 0, no occurrences looked for in the data   
Unparsing: When maxOccurs is 0, no occurrences looked for in the infoset or written

*16.1.2* (implicit):  
Parsing: When maxOccurs is 0, no occurrences looked for in the data   
Unparsing: When maxOccurs is 0, no occurrences looked for in the infoset or written

*16.1.4* (expression):  
Parsing: When dfdl:occursCount is 0, no occurrences looked for in the data

**5.10**. *Sections various.* [*https://redmine.ogf.org/issues/245*](https://redmine.ogf.org/issues/245)

The specification is changed to say that a 'fixed length' element with a non-zero length can never have an empty representation. The emptyValueDelimiterPolicy does not therefore apply and all occurrences of the element will have the fixed length. This is to prevent a catch 22 situation when initiators are present. Spec changes:

Add a definition of 'fixed length element' to the glossary in section 3, being an element of specified length where lengthKind is 'explicit' but length is not an expression, or lengthKind is 'implicit' (of simple type only). (Note that the spec also uses 'fixed length' for choice branches where choiceLengthKind is 'explicit')

Add a paragraph to section 9.2.2 which states that empty representation is not possible for fixed length elements with a non-zero length.

Add a clause in 9.4.2.2 and 9.4.2.3 so that the statement about optional occurrences says "If dfdl:emptyValueDelimiterPolicy **is in effect and** is not 'none'...", as the words today do not cover cases when it is ignored (no initiator or terminator).

Add a new paragraph to the property description of emptyValueDelimiterPolicy in section 12.2 "Ignored if the element is fixed length and length is not zero (no empty representation is possible)."

Update existing paragraph of the property description of emptyValueDelimiterPolicy in section 12.2 "It is a schema definition error if dfdl:emptyValueDelimiterPolicy **is in effect and** is set to 'none' or 'terminator' when the parent xs:sequence has dfdl:initiatedContent 'yes'."

Update existing paragraph of the property description of nilValueDelimiterPolicy in section 13.15 "It is a schema definition error if dfdl:nilValueDelimiterPolicy **is in effect and** is set to 'none' or 'terminator' when the parent xs:sequence has dfdl:initiatedContent 'yes'."

Update existing paragraph of the property description of nilKind in section 13.15 to remove the definition of fixed length.

**5.11**. *Section 23.4.* [*https://redmine.ogf.org/issues/246*](https://redmine.ogf.org/issues/246)

Clarify the meaning of an unqualified path step in a DFDL expression. Add new note after table 57 with accompanying reference.

5. NameTest - These QNames are path steps that refer to elements in the DFDL infoset. If such an element is in a namespace, then the NameTest QName must have a prefix which is bound to the namespace. Specifically, any default namespace is not used to implicitly qualify these NameTest QNames. This behavior is consistent with XPath expression usage in XML Schema [x] such as in the path property of the xs:selector and xs:field elements within xs:key and xs:unique constraints, and in related XML standards such as XSLT. Note however, that this behavior is different from the way QNames are used in other places in XML and DFDL Schemas such as the ref property of an element reference, or the dfdl:ref property of a DFDL format annotation. There a QName with no prefix must always be referring to a global declaration or definition, and so is augmented with the default namespace when needed.

[x] Definitive XML Schema (Walmsley, ISBN 0-13-065567-8) page 390, Section 17.8, Table 17-6 says "A child element-type name which must be prefixed if it is in a namespace".

**5.12**. *Section 12.3.7.2.1.* [*https://redmine.ogf.org/issues/247*](https://redmine.ogf.org/issues/247)

Correct the xs:decimal row in table 22 so that the minimum number of bits is 8. This is because lengthUnits 'bits' is not allowed for xs:decimal. A footnote is added to the table stating this restriction.

**5.13***. Section 13.7.1.1.* [*https://redmine.ogf.org/issues/248*](https://redmine.ogf.org/issues/248)

The paragraphs in section 13.7.1.1 are incorrect and should read as follows:

“For both parsing and unparsing, the bit string that represents the content region for a base-2 binary number is converted to/from an Infoset value by a calculation that involves the length and the dfdl:byteOrder and dfdl:bitOrder properties.

When parsing, DFDL specifies how an unsigned integer of unbounded magnitude is computed from a bit string based on its length, and the dfdl:byteOrder and dfdl:bitOrder properties. For signed types, this unbounded integer is converted into a signed value by way of the well-known twos-complement scheme, and for the xs:decimal type, the dfdl:binaryDecimalVirtualPoint property can be used to convert this integer into a decimal value with an integer and a fractional component

A DFDL implementation can use any conversion technique consistent with this description.”

**5.14**. *Section 13.7.* [*https://redmine.ogf.org/issues/249*](https://redmine.ogf.org/issues/249)

In the description of binaryDecimalVirtualPoint, clarify that when unparsing, if the property value is not sufficient to remove the decimal point from the infoset value, it is a processing error. This is true even if the resultant number can be converted into an integer (that is, all digits after the decimal point are zero) because it is an example of excess precision.

**5.15***. Section 12.3.* [*https://redmine.ogf.org/issues/253*](https://redmine.ogf.org/issues/253)

Correct the description of lengthUnits to state it can have the value ‘bits’ for packed calendars.

“- 'bits' may only be used for xs:boolean, xs:byte, xs:short, xs:int, xs:long, xs:unsignedByte, xs:unsignedShort, xs:unsignedInt and xs:unsignedLong simple types with binary representation, and for calendar simple types with binary packed representation. ”

**5.16***. Section 12.3.3, 12.3.4.* [*https://redmine.ogf.org/issues/254*](https://redmine.ogf.org/issues/254)

Correct table 18 and table 20 to reflect that binarySeconds and binaryMilliseconds are not allowable representations for types xs:date and xs:time.

**5.17***. Section 13.11.* [*https://redmine.ogf.org/issues/257*](https://redmine.ogf.org/issues/257)

The regular expression for calendarTimeZone has one extra closing bracket on the far right-hand end. The bracket is removed.

**5.18***. Section 3.11.* [*https://redmine.ogf.org/issues/258*](https://redmine.ogf.org/issues/258)

Erratum 2.100 in [DFDLX1] states:

*“State that when unparsing an element with lengthKind ‘explicit’ and where length is an expression, then the data in the Infoset is treated as variable length and not fixed length. The behaviour is the same as lengthKind ‘prefixed’.”*

The implication is that the length expression is ignored when unparsing. However IBM DFDL has an established behaviour where the length expression is evaluated when unparsing and the resultant length used.

Due to this, erratum 2.100 is reversed. The specification will revert to describing lengthKind 'explicit' as fixed length for both literal and expression lengths, and the length expression is evaluated when unparsing to provide the length to use.

The original variable length behaviour scenario that motivated erratum 2.100 can still be achieved using outputValueCalc with dfdl:valueLength() function.

**5.19**. *Section 16.1.1, 16.1.2.* [*https://redmine.ogf.org/issues/259*](https://redmine.ogf.org/issues/259)

Change the unparsing behaviour for occursCountKind ‘fixed’ and ‘implicit’ to state that the unparser stops looking for occurrences in the Infoset once maxOccurs has been reached. The spec currently states that it is a processing error if more than maxOccurs are found, which is not symmetric with parsing behaviour.

**5.20**. *Section 23.1 and others.* [*https://redmine.ogf.org/issues/260*](https://redmine.ogf.org/issues/260)

An outputValueCalc property expression may reference an element that follows the element that carries the property. Add that how far forward the expression can reference is implementation-defined.

The following properties or annotations need ‘no forward reference’ clauses adding to their descriptions: calendarLanguage, choiceDispatchKey, setVariable.

**5.21**. *Section 15.* [*https://redmine.ogf.org/issues/263*](https://redmine.ogf.org/issues/263)

Erratum 3.15 in [DFDLX1] introduced direct dispatch choice. The comparison between choiceDispatchKey and choiceBranchKey was originally case-insensitive, but this was changed by public comment to be case-sensitive, on performance grounds. The public comment update was not applied to the property description for choiceBranchKey. It is updated to say case-sensitive.

**5.22**. *Section 13.2.1.* [*https://redmine.ogf.org/issues/264*](https://redmine.ogf.org/issues/264)

Clarify that when a block escape scheme is in force for an element when parsing, it is a processing error if the end of the data for the element is reached and the escapeBlockEnd is not found in the data.

**5.23** *Section 13.16* [*https://redmine.ogf/org/issues/283*](https://redmine.ogf/org/issues/283)

nilKind, nilValue, and nilValueDelimiterPolicy properties apply to nillable complex type elements as well as simple types.

Change the last line in the description of each property to remove the “(simpleType)” qualifier so that it reads “Annotation: dfdl:element”.

**5.24** *Sections 13.11.1, 12.3.7.2.6.* [*https://redmine.ogf.org/issues/297*](https://redmine.ogf.org/issues/297)

Section 13.11.1 to be updated as follows:

1. Update the 'S' row in the table to strike "(see note 1)" and change '24' to '23'.
2. Move this paragraph to immediately below the table:  
   "The count of pattern letters determines the format as indicated in the table."
3. Add paragraph:  
   "When numeric fields abut one another directly, with no intervening delimiter characters, they constitute a run of abutting numeric fields. Such runs are parsed specially as described at [ICUDateTime]."
4. Make sentence into separate paragraph:  
   "Unlike other fields, fractional seconds "S" are padded on the right with zero."
5. Revise existing first paragraph below Table 40:  
   "The maximum number of "S" symbols that may appear in the pattern is implementation-defined, but must be at least three. The stored accuracy for fractional seconds is also implementation-defined, but must be at least millisecond accuracy. When the number of "S" symbols in a pattern exceeds the supported accuracy, excess fractional seconds are truncated from the right (not rounded) when parsing, and zeros are added to the right when unparsing. For example, a DFDL processor allows up to six "S" symbols and has millisecond accuracy; for pattern "ss.SSSSSS", data "12.345678" would be parsed into infoset xs:time "00:00:12:345", which would be unparsed into data "12.345000".
6. The fractional second bullet in the description of the I symbol should change

from “The number of fractional second digits supported is implementation-defined but must be at least millisecond accuracy.” to “The number of fractional second digits supported is the same as for the “S” fractional seconds specifier described above”.

Section 12.3.7.2.6 – Change “implementation-dependent” to “implementation-defined”.

**5.25** *Section 17.* [*https://redmine.ogf.org/issues/300*](https://redmine.ogf.org/issues/300)

In the description of the inputValueCalc property, remove the sentence: “This property is not allowed to appear on a local element or element reference that is the root of a choice branch.”

**5.26** *Section 2.* [*https://redmine.ogf.org/issues/304*](https://redmine.ogf.org/issues/304)

The sentence:

The key words must, must not, required, shall, shall not, should, should not, recommended, may, may not and optional in this document are to be interpreted as described in [[RFC2119](https://opensource.ncsa.illinois.edu/projects/artifacts/DFDL/latest/documentation/Manual/#a_RFC2119)].

is modified to:

The key words must, must not, shall, shall not, should, should not, recommended, may, and may not in this document are to be interpreted as described in [[RFC2119](https://opensource.ncsa.illinois.edu/projects/artifacts/DFDL/latest/documentation/Manual/#a_RFC2119)]. The terms *required* and *optional* in this document are used as described in Section 3 (Glossary).

**5.27** *Section 11.* [*https://redmine.org.org/issues/306*](https://redmine.org.org/issues/306)

Description of the bitOrder property says: "The bit order can only change on byte boundaries, and alignment of up to 7 bits will be inserted to ensure byte-alignment whenever the bit order changes." Should say: "The bit order can only change on byte boundaries, and alignment of up to 7 bits will be skipped (parsing) or inserted (unparsing) to ensure byte-alignment whenever the bit order changes."

**5.28** *Section 13.6.* [*https://redmine.ogf.org/issues/313*](https://redmine.ogf.org/issues/313)

The descriptions of properties textStandardNaNRep and textStandardInfinityRep should state empty string not allowed. It is a schema definition error if empty string found as the property value.

**5.29** *Sections 23.4, 23.5.3.* [*https://redmine.ogf.org/issues/314*](https://redmine.ogf.org/issues/314)

For testing a value is in a set, add to Section 23.4:

Add XPath 2.0 'intersect' and "except" operators to the list of supported operators.   
MultiplicativeExpr ::= IntersectExceptExpr ( ("\*" | "div" | "idiv" | "mod") IntersectExceptExpr )\*   
IntersectExceptExpr ::= UnaryExpr ( ("intersect" | "except") UnaryExpr )\*

For testing a value is in a range, add to Section 23.5.3 two DFDL functions:  
dfdl:checkRangeInclusive($node, $val1, $val2)  
dfdl:checkRangeExclusive($node, $val1, $val2)  
Returns boolean true if the specified node value is in the range given by $val1 and $val2.   
The type of $val1 and $val2 must be compatible with the type of $node, and must be a derivative of xs:decimal, xs:float or xs:double. It is a schema definition error if the $node argument is a complex element.

**5.30** ***Section 12.3.2.2.***[*https://redmine.ogf.org/issues/315*](https://redmine.ogf.org/issues/315)

Add the sentence:

Implementation Note: Scanning for delimiters when data is binary, or when using raw byte entities in delimiters, means that a simple character based delimiter scanner is not sufficient, as the delimiter may not be representable as characters.

**5.31** *Section 9.5.* [*https://redmine.ogf.org/issues/322*](https://redmine.ogf.org/issues/322)

Update “For elements and element refs”:

* replace “refs” with “references”.
* Item 2 update to “2. dfdl:element following property scoping rules and evaluating any property expressions (corresponds to SimpleElement grammar region)

Update “For sequences, choices, and group refs”

* replace “refs” with “references”.
* Item 4 update to: “4. dfdl:sequence or dfdl:choice or dfdl:group following property scoping rules evaluating any property expressions (corresponds to ComplexContent grammar region)

**5.32** *Section 9.3.2.1* [*https://redmine.ogf.org/issues/309*](https://redmine.ogf.org/issues/309)

Update bullet 1 which says “1. nil representation (if %ES; is a literal nil value).” to say “1. nil representation (if %ES; or %WSP\*; on its own is a literal nil value).”

Also Section 9.2.5 the phrase

“… can be a zero-length representation if dfdl:nilValue is "%ES;",...”

should be replaced with

"... can be a zero-length representation if %ES; or %WSP\*; on its own is a literal nil value, and ...."

**5.33** *Section 13.9 https://redmine.ogf.org/issues/312*

For property textBooleanTrueRep, the sentence “It is a schema definition error if the string literal contains any of the disallowed constructs.” is extended to say “It is a schema definition error if the string literal is the empty string, or contains any of the disallowed constructs.”

**5.34** *Section 13.2 https://redmine.ogf.org/issues/320*

In the description of property textPadKind, the sentence "The padding characters populate the LeftPadding and/or RightPadding regions depending on dfdl:textStringJustification, dfdl:textNumberJustification, or dfdl:textCalendarJustification, depending on the type of the element." omits the dfdl:textBooleanJustification case.

The proper phrasing to match the wording of the prior sentence is

"The padding characters populate the LeftPadding and/or RightPadding regions depending on dfdl:textStringJustification, dfdl:textNumberJustification, dfdl:textBooleanJustification, or dfdl:textCalendarJustification, depending on the type of the element."

**5.35** *Section 5.2 https://redmine.ogf.org/issues/321*

In the bullet item that says

* pattern (for type xs:string and all types descending from xs:string in Figure 3 DFDL simple types)

The wording in parenthesis is dropped.

**5.36** *Section 15 https://redmine.ogf.org/issues/326*

The definition of the choiceBranchKey property is changed from “DFDL String Literal” to “List of DFDL String Literals”.

The phrase “When the dfdl:choiceDispatchKey expression evaluates to a string matching this property’s value…” is changed to say “When the dfdl:choiceDispatchKey expression evaluates to a string matching one of this property’s values….”

In the definition of the choiceDispatchKey property, the phrase “The resultant string must match the dfdl:choiceBranchKey property value of one of the ….” Becomes “The resultant string must match one of the dfdl:choiceBranchKey property values of one of the …”. Also the phrase “… does not match one of the dfdl:choiceBranchKey properties for the branches.” becomes “… does not match any of the dfdl:choiceBranchKey property values for the branches.”

Corresponding changes must be made in other sections that describe choiceBranchKey matching, which are:

* Section 9.3.1.1 “… matches the dfdl:choiceBranchKey property of the child.” Becomes “… matches one of the dfdl:choiceBranchKey values of the child.”
* Section 9.3.1.3 “… does not match the dfdl:choiceBranchKey property of the child.” Becomes “…” does not match any of the dfdl:choiceBranchKey values of the child.”
* Section 9.3.3 “ … to the value of the dfdl:choiceBranchKey property…” becomes “… to the value of one of the dfdl:choiceBranchKey property values …”
* Section 15.1.2 “…the string matching (case sensitive) the dfdl:choiceBranchKey property of one of the choice branches.” Becomes “… the string matching (case sensitive) one of the dfdl:choiceBranchKey values for one of the choice branches.” Also the phrase “It must be unique across all branches…” becomes “All values of dfdl:choiceBranchKey properties must be unique across all branches…”

**5.37** *Section 15 https://redmine.ogf.org/issues/328*

In the description of the choiceDispatchKey property, a sentence is added saying “It is a schema definition error if the expression contains forward references to elements which have not yet been processed.”

**5.38** *Section 4.1.2 https://redmine.ogf.org/issues/330*

An additional member is added to the list of members that follow the phrase “An element information item has the following members:” The new member is added after the [nilled] member. The description is:

**[array]** Boolean. True if the item is an array, meaning that it corresponds to an element having maxOccurs value greater than 1, or ‘unbounded’.

In addition, Figure 1, “DFDL Infoset Object Model” is modified to list the new member on the Element class. (This class is also missing the [nilled] Boolean, [valid] Boolean, and [unionMemberSchema] string members. The Document class is missing the [unicodeByteOrderMark] and [schema] members. All the missing members should be added along with the new ‘array’ member.)

**5.[3](https://redmine.ogf.org/issues/196)9** Section 15.1.1 Choice branches that are zero-occurrences arrays. (https://redmine.ogf.org/issues/299)

Section 15.1.1 is updated to add a numbered item 5.

5. If the branch is an element declaration having dfdl:occursCountKind='expression' or dfdl:occursCountKind='parsed', then zero instances are possible. If the branch parses successfully but produces no element occurrences, then the branch is considered missing, and the parser looks for the next branch.

**5.40**. Section 14.1 Empty Sequence & Hidden Group References (<https://redmine.ogf.org/issues/282>)

Section 14.1 is updated with this additional paragraph

A hidden group reference is indicated in DFDL using an empty sequence such as <xs:sequence dfdl:hiddenGroupRef="QName"/>. To XML Schema this is an empty sequence group; hence it is an SDE if this appears as the model group of a complex type. Otherwise this is not considered an empty sequence, but a group reference.

**5.41** Section 2.6 Revised use of RFC 2119 words must, should, etc. https://redmine.ogf.org/issues/304

Section 2.6: "...then a Dynamic Type Error should cause a Schema Definition Error" change the term "should" to "must".

***(Additional entries expected here. This erratum is not complete yet. )***

**5.42** Section 21: Optional Features – add dfdl:utf16Width property. (https://redmine.ogf.org/issues/337)

The table entry should appear below “Extended Encodings” in the table for relevance to text encodings.

**5.43** Remove bidirectional text properties. (<https://redmine.ogf.org/issues/357>)

These will be proposed as experimental feature for a future version of the DFDL spec.

Sections affected are: 6.3.1, 21 (search for “bi-directional” note that there is one usage of this term that is not with reference to bidi text), 4.1.2 [data-value], 6.3.1.1, 13.3, 22.1.1, 22.2.1(search for “bidi”)

**5.44** Add ComplexContent production to Data Syntax Grammar (<https://redmine.ogf.org/issues/316>)

This was Action 242 Part 1. For rationale, see the tracker.

Section 9.2, The grammar is changed as follows:

ComplexNormalRep = LeftFraming PrefixLength ComplexContent RightFraming

ComplexContent = ComplexValue ElementUnused

ComplexValue = Sequence | Choice

Other Sections affected:

* 12.3.3 “For complex elements, 'implicit' means the length is determined by the combined lengths of the contained children, that is the ComplexContent region. However, note that alignment regions inside the ComplexContent region may be of different lengths depending on the ComplexContent's starting position alignment.” Changed to: “For complex elements, 'implicit' means the length is determined by the combined lengths of the contained children, that is the ComplexValue region, and the ElementUnused region is of size 0. However, note that alignment regions inside the contained children within the ComplexValue region may be of different lengths depending on the ComplexValue's starting position alignment.
* 23.5.3 for function dfdl:valueLength, the descriptive text use of ComplexContent changes to ComplexValue.

**5.45** Clarifications of grammar and SimpleContent vs. SimpleValue contents. (<https://redmine.ogf.org/issues/317>)

This was Action 242 part 2. For rationale, see the tracker.

Section 9.2 Data Syntax Grammar is modified as follows:

New rule: NilLiteralValue = SimpleNilLiteralValue | ComplexNilLiteralValue

Rename: SimpleValue to SimpleNormalValue

New rule: SimpleLogicalValue = SimpleNormalValue | NilLogicalValue

Section 13.2.1

* escapeCharacter – the sentence “Escape characters contribute to the content length of the field” is changed to “Escape characters contribute to the simple value region (SimpleLogicalValue or SimpleNilLiteralValue) of the field.”
* escapeBlockStart, escapeBlockEnd – similarly change their mention of “content length” as for escapeCharacter
* escapeEscapeCharacter – add sentence: “Escape-escape characters contribute to the simple value region (SimpleLogicalValue or SimpleNilLiteralValue) of the field.”
* extraEscapedCharacters – add sentence “Extra escaped characxters contribute to the simple value region (SimpleLogicalValue or SimpleNilLiteralValue) of the field.”

Section 23.5.3

* for dfdl:valueLength change “SimpleValue or NilLogicalValue” to “SimpleLogicalValue”.

**5.46** Circular Deadlocks (https://redmine.ogf.org/issues/318) (For rationale, see the tracker.)

Add new Section

23.6 Unparsing and Circular Expression Deadlock Errors

It is possible for expressions and lengths of elements in a DFDL schema to interact badly, resulting in circular deadlocks where an expression is unable to evaluate because it depends in some way on the length of something that depends on the expression itself.

Expression deadlocks are always schema definition errors.

One scenario where such a deadlock can arise is due to what is called the i*nterior-alignment problem*.

This problem arises due to this observation: the length of a complex element may vary depending on the alignment of its start. This is because the ComplexValue region contains child elements, and those elements can have alignments specified. Hence, the alignment regions for those child elements can vary in size depending on their starting position. Those alignment regions contribute to the ComplexValue region length of the enclosing complex element, making it variable length depending on its starting position.

In many data formats, there are length elements which precede a complex element. The value of such elements is used to compute the length when parsing. When unparsing, these preceding length elements are often computed using dfdl:outputValueCalc expressions. Those expressions may invoke the dfdl:valueLength() function on an element of complex type. This expression is forward referencing to parts of the data that have yet to be unparsed; hence, their length has not yet been computed. A DFDL unparser must detect this and suspend unparsing of the preceding length element until the forward reference can be resolved.

While it is not common in many data formats, DFDL allows for this preceding length element to itself be a variable length element, such as a number with dfdl:representation ‘text’ and dfdl:lengthKind ‘delimited’.

In this case, the length of this preceding length element cannot be determined until the value of the dfdl:outputValueCalc expression has been computed.

If that dfdl:outputValueCalc expression depends on the valueLength of a following complex element which due to interior alignments, has a length that depends on its starting position, then we have a circular deadlock. The length of the complex element depends on the starting position, which depends on the length of the preceding length element, which depends on the value of the dfdl:outputValueCalc expression, which depends on the length of the following complex element.

This scenario is possible due to the way that language features compose in DFDL. The design intention in DFDL is for the various language features to be fairly orthogonal to each other. Alignment causes representation lengths to break this orthogonality because length of one thing now depends on the interior alignment and hence, the length of previous things.

Fortunately, elements with dfdl:outputValueCalc are rarely variable length.

**5.47** Detection of Encoding/Decoding Errors (<https://redmine.ogf.org/issues/319>)

(For Rationale, see the tracker.)

Section 11.2.1.1 additional paragraphs are added:

Detection of character set decoding errors is often implementation-dependent because

DFDL Implementations are free to optimize processing speed by skipping character decoding or encoding whenever possible. For example: When character set encodings are fixed-width, it is possible to determine lengths in bytes or bits from the length in characters by multiplying the length value by the character width, without having to decode any characters.

When parsing, character decoding errors MUST be detected when

* 1. the decoding results in a character being placed into the DFDL Infoset
  2. the decoding is necessary to identify a delimiter
  3. the decoding is necessary to determine a match or non-match of a regular expression in a dfdl:assert or dfdl:discriminator with testKind=’pattern’.

When unparsing, character encoding errors MUST be detected when

1. an unmapped character appears in the infoset value of an element.

In all other cases, character set decoding and encoding errors MAY NOT be detected.

**5.48** fn:error() function (<https://redmine.ogf.org/issues/324>)

New Section 23.5.5 Miscelaneous Functions

|  |  |
| --- | --- |
| **Function** | **Meaning** |
| fn:error()  fn:error($id as xs:string)  fn:error($id as xs:string,  $desc as xs:string,  $obj?) | Causes a processing error.  This function does not return a value. A processing error ends the evaluation of the expression.  The $id argument is an error code identifier string that distinguishes this error from others. The string should have the structure of an XSD QName; the namespace URI conventionally identifies the component, subsystem, or authority responsible for defining the meaning of the error code, while the local part identifies the specific error condition. This information is incorporated into any diagnostic messages created by the DFDL implementation in response to the processing error in an implementation-dependent manner. If the $id argument string does not have the form of an XSD QName, or the QName cannot be interpreted as a meaningful namespace prefix and local identifier, then the processing error still occurs but the diagnostic message is created in an implementation-dependent manner.  The $desc is a natural-language description of the error condition. This string will appear in any diagnostic messages created by the DFDL implementation in response to the processing error.  The $obj? argument is an arbitrary value used to convey additional information about the error and it is used to construct the diagnostic message in an implementation-dependent manner.  If any argument is not supplied the processing error occurs but the diagnostic message created is implementation-dependent. |

**5.49** Allow import of other annotation language schemas (<https://redmine.ogf.org/issues/334>)

New Section 5.3 Compatibility with Other Annotation Language Schemas

A DFDL Schema only applies annotations on a subset of the XML Schema constructs. However, a DFDL schema may be annotated not only by DFDL annotations, but by other XML annotation languages.

The XML schemas of those other annotation languages MAY use any constructs of XML Schema, including those prohibited by DFDL (such as attribute declarations.)

A DFDL implementation MUST ignore any schema file included or imported by a DFDL schema if the top level xs:schema element of that schema does not have an XML namespace binding for the DFDL namespace.

**5.50** Remove BOM Processing (<https://redmine.ogf.org/issues/342>)

Section changes:

* 4.1.1. Remove [unicodeByteOrderMark] enum from the infoset
* 9.2. Remove unicodeByteOrderMark from the grammar.
* 11. Remove forward reference to 11.1 from the 'Encoding' property description.
* 11.1. Remove section and two footnotes.

**5.51** %ES: allowed alone in terminator when not delimited format (<https://redmine.ogf.org/issues/343>)

Section 12.2 The language "ES must not appear as the only DFDL string literal in the property. It can only appear as a member of a list." should be dropped. This phrase appears in the description of the initiator property and the terminator property and for consistency should be dropped from both places.

**5.52** choiceDispatchKey evaluating to empty string is SDE (<https://redmine.ogf.org/issues/352>)

Section 15 choiceDispatchKey property change “The expression must evaluate to an xs:string which must not be the empty string.” to "The expression must evaluate to an xs:string. It is a Schema Definition Error if the expression returns an empty string."

1. Security Considerations

Security considerations are dealt with in the corresponding sections of the DFDL 1.0 specification [DFDL].

No additional security issues have been raised.

1. Contributors

Stephen M. Hanson,

IBM Software Group,

Hursley,

Winchester,UK

[smh@uk.ibm.com](mailto:smh@uk.ibm.com)

Michael J. Beckerle,

Tresys Technology,

Columbia, MD, USA

[mbeckerle@tresys.com](mailto:mbeckerle@tresys.com)

We greatly acknowledge the contributions made to this document by the following people.

Tim Kimber, IBM Software Group, Hursley, UK

Alex Wood, IBM Software Group, Hursley, UK

Mark Frost, IBM Software Group, Hursley, UK

Andrew Edwards, IBM Software Group, Hursley, UK

1. Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

1. Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

1. Full Copyright Notice

Copyright (C) Open Grid Forum (2014-2018). Some Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included as references to the derived portions on all such copies and derivative works. The published OGF document from which such works are derived, however, may not be modified in any way, such as by removing the copyright notice or references to the OGF or other organizations, except as needed for the purpose of developing new or updated OGF documents in conformance with the procedures defined in the OGF Document Process, or as required to translate it into languages other than English. OGF, with the approval of its board, may remove this restriction for inclusion of OGF document content for the purpose of producing standards in cooperation with other international standards bodies.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

1. References

[DFDL] OGF DFDL 1.0 specification

<http://www.ogf.org/documents/GFD.207.pdf/>

[GFD] OGF Document Process and Requirements <http://www.ogf.org/documents/GFD.152.pdf/>

[XPATH2] XPath 2.0

http://www.w3.org/TR/xpath20/

[ISSUES] OGF Redmine Issue Trackers

<http://redmine.ogf.org/projects/dfdl-wg/issues>

[DFDLX1] DFDL Experience Document 1(OBSOLETE)

<https://www.ogf.org/documents/GFD.214.pdf>