OpenGridForum

OPEN FORUM | OPEN STANDARDS

Data Format Description Language (DFDL)

DFDL 1.0 – Proposed Recommendation

Steve Hanson, IBM (smh@uk.ibm.com)

Admin

- DFDL WG co-chairs:
  - Steve Hanson, IBM UK
  - Mike Beckerle, Oco Inc
- Two note takers please
- Sign the attendance sheet
- Note: OGF Intellectual Property Rules apply

OGF IPR Policies Apply

- "I acknowledge that participation in this meeting is subject to the OGF Intellectual Property Policy."
- Intellectual Property Notices Note Well: All statements related to the activities of the OGF and addressed to the OGF are subject to all provisions of Appendix B of GFD-C.1, which grants to the OGF and its participants certain licenses and rights in such statements. Such statements include verbal statements in OGF meetings, as well as written and electronic communications made at any time or place, which are addressed to:
  - the OGF plenary session,
  - any OGF working group or portion thereof,
  - the OGF Board of Directors, the GFSG, or any member thereof on behalf of the OGF,
  - the ADCOM, or any member thereof on behalf of the ADCOM,
  - any OGF mailing list, including any group list, or any other list functioning under OGF auspices,
  - the OGF Editor or the document authoring and review process
- Statements made outside of a OGF meeting, mailing list or other function, that are clearly not intended to be input to an OGF activity, group or function, are not subject to these provisions.
- Excerpt from Appendix B of GFD-C.1: "Where the OGF knows of rights, or claimed rights, the OGF secretariat shall attempt to obtain from the claimant of such rights, a written assurance that upon approval by the GFSG of the relevant OGF document(s), any party will be able to obtain the right to implement, use and distribute the technology or works when implementing, using or distributing technology based upon the specific specification(s) under openly specified, reasonable, non-discriminatory terms. The working group or research group proposing the use of the technology with respect to which the proprietary rights are claimed may assist the OGF secretariat in this effort. The results of this procedure shall not affect advancement of document, except that the GFSG may defer approval where a delay may facilitate the obtaining of such assurances. The results will, however, be recorded by the OGF Secretariat, and made available. The GFSG may also direct that a summary of the results be included in any GFD published containing the specification."

- OGF Intellectual Property Policies are adapted from the IETF Intellectual Property Policies that support the Internet Standards Process.

Full Copyright Notice

Agenda

OpenGridForum

- Why DFDL?
  - The problem it solves
- What is DFDL?
  - Quick overview for newbies
- DFDL 1.0 status
  - Specification & implementations

- Backup: DFDL Language

Agenda

OpenGridForum

- Why DFDL?
  - The problem it solves
- What is DFDL?
  - Quick overview for newbies
- DFDL 1.0 status
  - Specification & implementations
- Backup: DFDL Language

Why DFDL?

OpenGridForum

- Much of the data in the world
  - Resides in files
  - Is not XML
  - Is a mixture of textual and binary
  - Has custom syntax and encodings
  - Does not have a shareable description

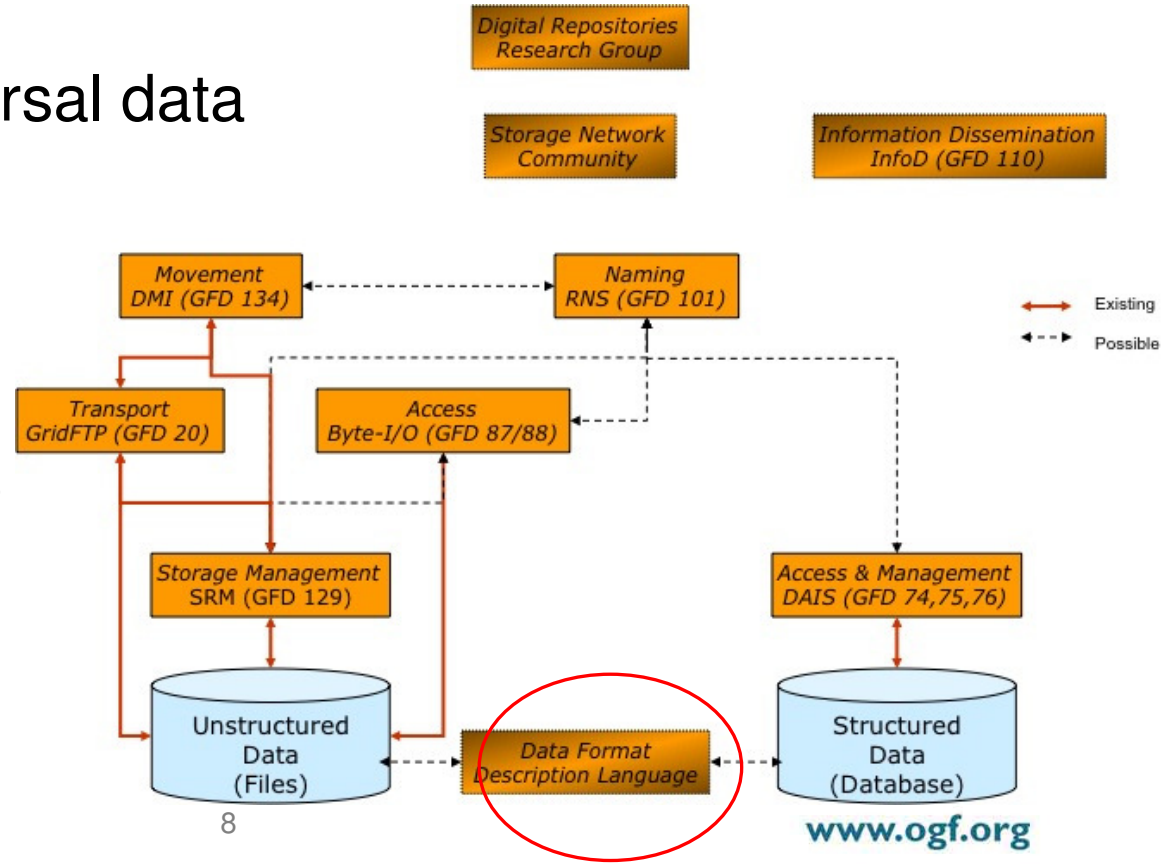
- Existing standards are not flexible enough
  - Prescriptive: “Put your data in this format!”
  - Textual – XML, JSON, EDI
  - Binary – ASN.1, XDR, EBML, …
  - You use the defined encodings, syntax, …
  - But descriptions are shareable

✓ **DFDL:** a universal, shareable, description for any data format

Grids and DFDL

- Grids are about big-data and big-computation problems
  - Simplistic solutions like “use XML” won’t cut it!
  - Performance and space usage

- Grids are about universal data interchange
  - XML
    - use XML Schema
  - Relational
    - use RDBMS schema
  - Other
    - ✓ use **DFDL Schema**



Agenda

OpenGridForum

- Why DFDL?
  - The problem it solves
- What is DFDL?
  - Quick overview for newbies
- DFDL 1.0 status
  - Specification & implementations
- Backup: DFDL Language

What is DFDL?

OpenGridForum

1. A way of *describing* data…
   - It is NOT a data format itself!
2. That can describe any data format…
   - Textual and binary
   - Commercial and scientific
   - Modern and legacy
   - Fixed length and delimited
3. While allowing high performance…
   - Choose the right data format for the job
   - High density, optimized I/O, random access
   - No need to use DFDL libraries

Benefits of DFDL

OpenGridForum

1. Data format independent
2. Prescriptive standards not always best for job
3. Descriptions can be shared
4. Interoperability
5. Allows after-the-event description
6. Supports move to SOA
7. Spans commercial and scientific worlds
8. Supports grid and cloud computing

DFDL Goals

OpenGridForum

- Build on existing standards
  - Leverage XML technology and concepts
- Support very efficient parsers/serializers
- Support round-tripping
  - Read and write data in described format from same description
- Keep simple cases simple
  - Simple descriptions should be "human readable"
- Generality
  - Can describe any data format
  - Allow extensions for new data formats

XML Synergy

- Use XML Schema subset & type system to describe the ***logical*** format of the data
- Use annotations within the XSD to describe the ***physical*** representation of the data
- Use XPath when referencing fields within the data

- Same approach actively used today:
  - IBM WebSphere Message Broker
  - Microsoft BizTalk flat file
  - Others

What DFDL is Not: FAQ

OpenGridForum

- I have a pre-defined XML Schema.
- Q: Can I use DFDL to populate it from a non-XML data file?

- A: Only partly:
  - DFDL is focused on data *format*
  - DFDL does *not* provide general data transformation
  - Populating a pre-defined XML Schema involves two separate problems:
    1. Using DFDL to describe the data file format
    2. Using a transformation system to transform that to conform to the pre-defined schema (not DFDL's job)

DFDL is only about Format !

OpenGridForum

- The structure of the DFDL schema is dictated by the logical structure of the data

- You must work bottom up.
- Start from the data format, not from what you want to turn it into.

DFDL Features

Release 1.0:

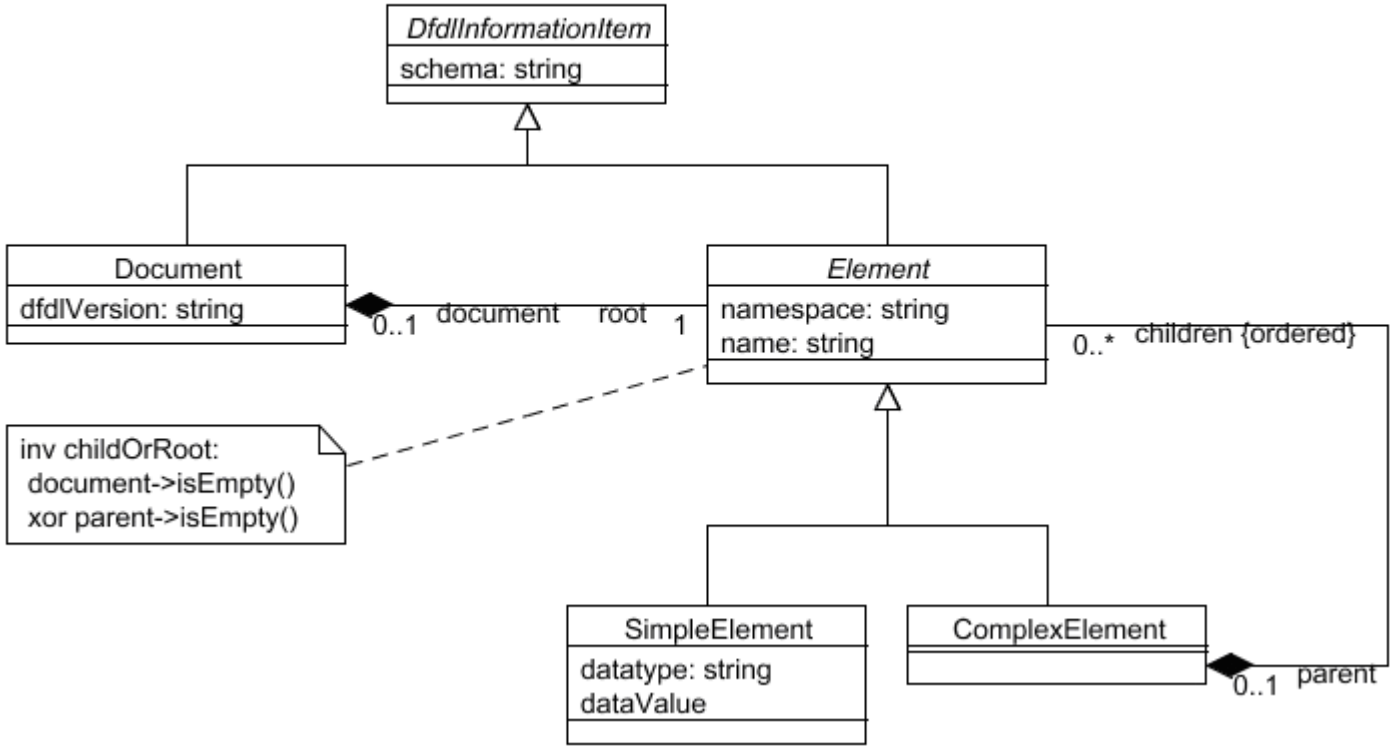
- Subset of XML Schema 1.0
- Rich textual & binary data capabilities including bit support
- Scoping rules that govern how the annotations apply
- Validated input and output - from XML Schema
- Defaults - for missing values
- ‘Nil’ capability - for out-of-band values
- Reference – use of a previously read value in subsequent expressions
- Expression language including variables
- Uncertainty – stratagems to resolve choices, optionality
- Arrays – one dimension
- Very general parsing/writing capability

Future:

- Multi-layer – intermediate representation not exposed in final result
- Extensibility – new type/transform specification
- Arrays - multi-dimension

DFDL Information Set

- An abstract data set defining the content that must be provided
  - To an application by a DFDL parser
  - To a DFDL unparser by an application
- Similar to XML Data Model (XDM)



DFDL Specification Conformance

OpenGridForum

- We want to make it easier to create conforming DFDL processors

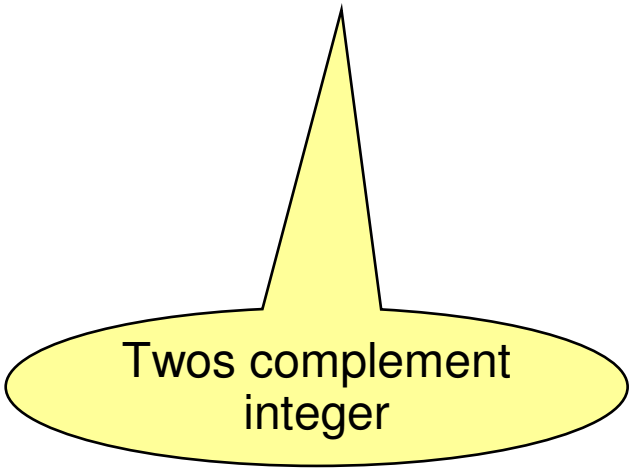
- To assist, the specification allows omission of:
  1. DFDL serializer
  2. Optional features

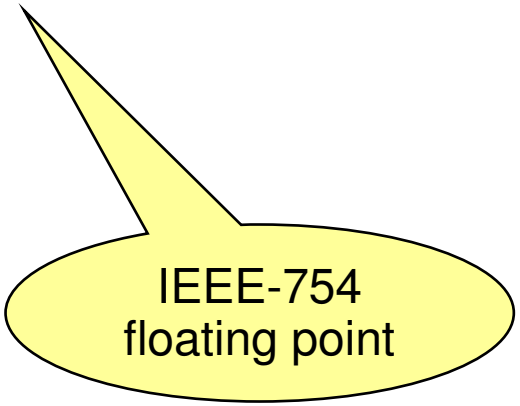
- Three levels of conformance for a DFDL processor with respect to optional features:
  - Minimal
  - Extended
  - Full

Example: Binary Data

OpenGridForum

```
0000 0005 ce29 46f6
```





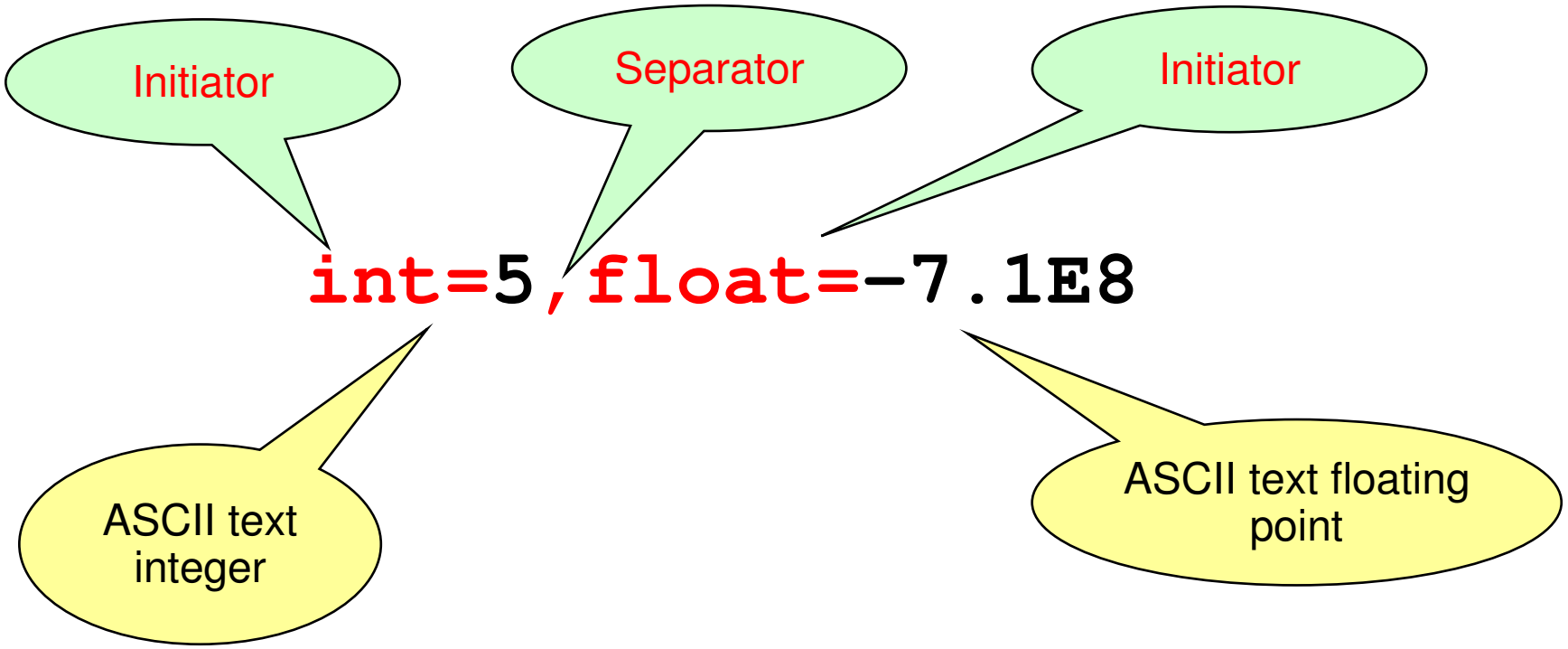
Example DFDL: Binary Data

```
<xs:complexType name="myBinary">
  <xs:sequence>
    <xs:element name="myInt" type="xs:int">
      <xs:annotation>
        <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
          <dfdl:element representation="binary"
              binaryNumberRep="binary" byteOrder="bigEndian"
              lengthKind="explicit" length="4"/>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="myFloat" type="xs:float">
      <xs:annotation>
        <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
          <dfdl:element representation="binary"
              binaryFloatRep="ieee" byteOrder="bigEndian"
              lengthKind="explicit" length="4"/>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

DFDL annotation

DFDL properties

Example: Textual Data



Separators, initiators, terminators are all examples in DFDL of *delimiters*

Example DFDL: Textual Data

OpenGridForum

```
<xs:complexType name="myText">
  <xs:sequence>
    <xs:annotation>
      <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
        <dfdl:sequence separator="," encoding="ascii"/>
      </xs:appinfo>
    </xs:annotation>
    <xs:element name="myInt" type="xs:int">
      <xs:annotation>
        <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
          <dfdl:element representation="text"
             textNumberRep="standard" encoding="ascii"
             lengthKind="delimited" initiator="int=" …/>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="myFloat" type="xs:float">
      <xs:annotation>
        <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
          <dfdl:element representation="text"
             textNumberRep="standard" encoding="ascii"
             lengthKind="delimited" initiator="float=" …/>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

Example DFDL: Short Form


```
<xs:complexType name="myText">
  <xs:sequence dfdl:separator="," dfdl:encoding="ascii" >
    <xs:element name="myInt" type="xs:int"
                dfdl:representation="text"
                dfdl:textNumberRep="standard" dfdl:encoding="ascii"
                dfdl:lengthKind="delimited" dfdl:initiator="int=" … />
    <xs:element name="myFloat" type="xs:float"
               dfdl:representation="text"
                dfdl:textNumberRep="standard" dfdl:encoding="ascii"
                dfdl:lengthKind="delimited" dfdl:initiator="float=" … />
  </xs:sequence>
</xs:complexType>
```

Agenda

- Why DFDL?
  - The problem it solves
- What is DFDL?
  - Quick overview for newbies
- DFDL 1.0 status
  - Specification & implementations
- Backup: DFDL Language

History

- 2003: Initial DFDL proposal at GGF9
- 2004: XML Schema based approach
- 2004: IBM joins WG at GGF12
- 2004: Prototypes started
- 2007: Progress report at OGF 20
- 2008: Progress report at OGF 21
- 2009: Full-time author
- 2009: IBM implementation started
- 2010: Present specification at OGF 28 & 29
- 2010: OGF Public Comment phase
- 2010: NCSA implementation
- *2010: OGF Proposed Recommendation*

Status

- Spec is currently at draft 044
  - https://forge.gridforum.org/sf/go/doc16073?nav=1
- Process
  - ✓ Editor & AD review
  - ✓ Standards Council review
  - ✓ Public Comment stage
  - ✓ Respond to comments
  - ✓ Standards Council review ***← we are here***
  - *Final Editor review*
  - *Proposed Recommendation*

Public Comment Stage Updates

- Optional features and Minimal, Extended and Full conformance
- Changes to some property names
- Limitations on forward references
- Fixed length choices
- Rewrite of ‘nils and defaults’ section
- Properties nilIndicatorPath/Index dropped
- Packed and BCD representations may be delimited
- Leading/Trailing skip region may be specified in bits or bytes
- Alignment units unrestricted
- New DFDL functions for test/set bits
- Clarification of schema definition error scenarios
- Formally add PrefixLength region to grammar and list constraints
- Allow UTF-16 encoding to be fixed or variable width
- Revise assert and discriminator syntax and behaviour
- Assert and discriminator can use regular expressions
- Assert and discriminator can use schema facets
- Formalize regular expression language to be Java or Perl
- Simplify syntax for hidden elements
- More rounding options for text decimals
- Calculated values only apply to simple elements

Optional Features

OpenGridForum

Feature	Detection
Validation	External switch
Simple type restrictions	xs:simpleType in xsd
Nils	xs:nillable='yes' in xsd
Defaults	xs:default or xs:fixed in xsd
Bi-Directional text.	dfdl:textBiDi='yes'
Lengths in Bits	dfdl:alignmentUnits='bits' or dfdl:lengthUnits='bits'
Delimited lengths and representation binary	dfdl:representation='binary' (or implied) and dfdl:lengthKind='delimited'
Regular expressions	dfdl:lengthKind='pattern', dfdl:assert testkind 'pattern', dfdl:discriminator testkind 'pattern'
Zoned numbers	dfdl:textNumberRep='zoned'
Packed numbers	dfdl:binaryNumberRep='packed'
Packed calendars	dfdl:binaryCalendarRep='packed'
S/390 floats	dfdl:binaryFloatRep='ibm390Hex'
Unordered sequences	dfdl:sequenceKind='unordered'
Floating elements	dfdl:floating='yes'

Feature	Detection
DFDL functions in expression language	dfdl: functions in expression
Hidden groups	dfdl:hiddenRef <> ''
Calculated values	dfdl:inputValueCalc <> '' or dfdl:outputValueCalc <> ''
Escape schemes	dfd:defineEscapeScheme in xsd
Extended encodings	Any dfdl:encoding value beyond the core list
Asserts annotations	dfdl:assert in xsd
Discriminators annotations	dfdl:discriminator in xsd
Prefixed lengths	dfdl:lengthKind='prefixed'
Variables	dfdl:defineVariable, dfdl:newVariableInstances, dfdl:setVariable Variables in DFDL expression language

Implementations

OpenGridForum

- Early prototypes (out of date)
  - PNNL/NCSA ‘Defuddle’
    - http://defuddle.pnl.gov/
  - IBM ‘Virtual XML Garden’
    - http://www.alphaworks.ibm.com/tech/virtualxml
- 1.0 specification
  - IBM DFDL
    - Internal IBM implementation at the moment
    - Full conformance, parser & serializer
    - Intend to make test cases public
  - NCSA ‘Daffodil’
    - Extended conformance, parser only
    - Possible reference implementation ?

NCSA 'Daffodil'

- Implemented at the National Center for Supercomputing Applications, at the University of Illinois, Urbana-Champaign, USA
- 'Daffodil v1' will be released this year as open source
- For more information, please see:
  Rodriguez, Alejandro and Robert E. McGrath, *Daffodil: A New DFDL Parser. NCSA, 2010.*
  http://cet.ncsa.illinois.edu/publications/Daffodil-ANewDFDLParser.pdf
- Thanks for substantial support from the U.S. National Archives and Records Administration (NARA)

This work was supported through National Science Foundation Cooperative Agreement NSF OCI 05-25308 and Cooperative Support Agreements NSF OCI 04-38712 and NSF OCI 05- 04064by the National Archives and Records Administration.

NCSA 'Daffodil'

OpenGridForum

- Built using SCALA v2.8
  - Functional language that compiles to Java byte code
- DFDL parser generator
  - Takes a DFDL schema
  - Generates a DFDL parser
- Parser creates a DOM tree
  - Implemented in JDOM
- Implements 90% of specification
  - Extended conformance, no serializer

Tutorial

OpenGridForum

- Easier, non-normative way to learn DFDL
  - Same idea as XML Schema 1.0 Primer
- Divided into example-based Lessons so you can learn at your own pace
  - Lesson 1: Introduction
  - Lesson 2: Language Basics
  - Lesson 3: DFDL Properties
  - …
- Drafts available for Lessons 1 to 3

Next Steps

- Working Group meetings will continue, as there is still work to do!
  - Create OGF hosted web pages
  - Complete Tutorial
  - Make ‘Daffodil’ into freely available reference implementation
  - Define certification process
- Contributions welcome!
  - Join the mailing list
    - http://www.ogf.org/mailman/listinfo/dfdl-wg
  - Attend the Wednesday phone conferences

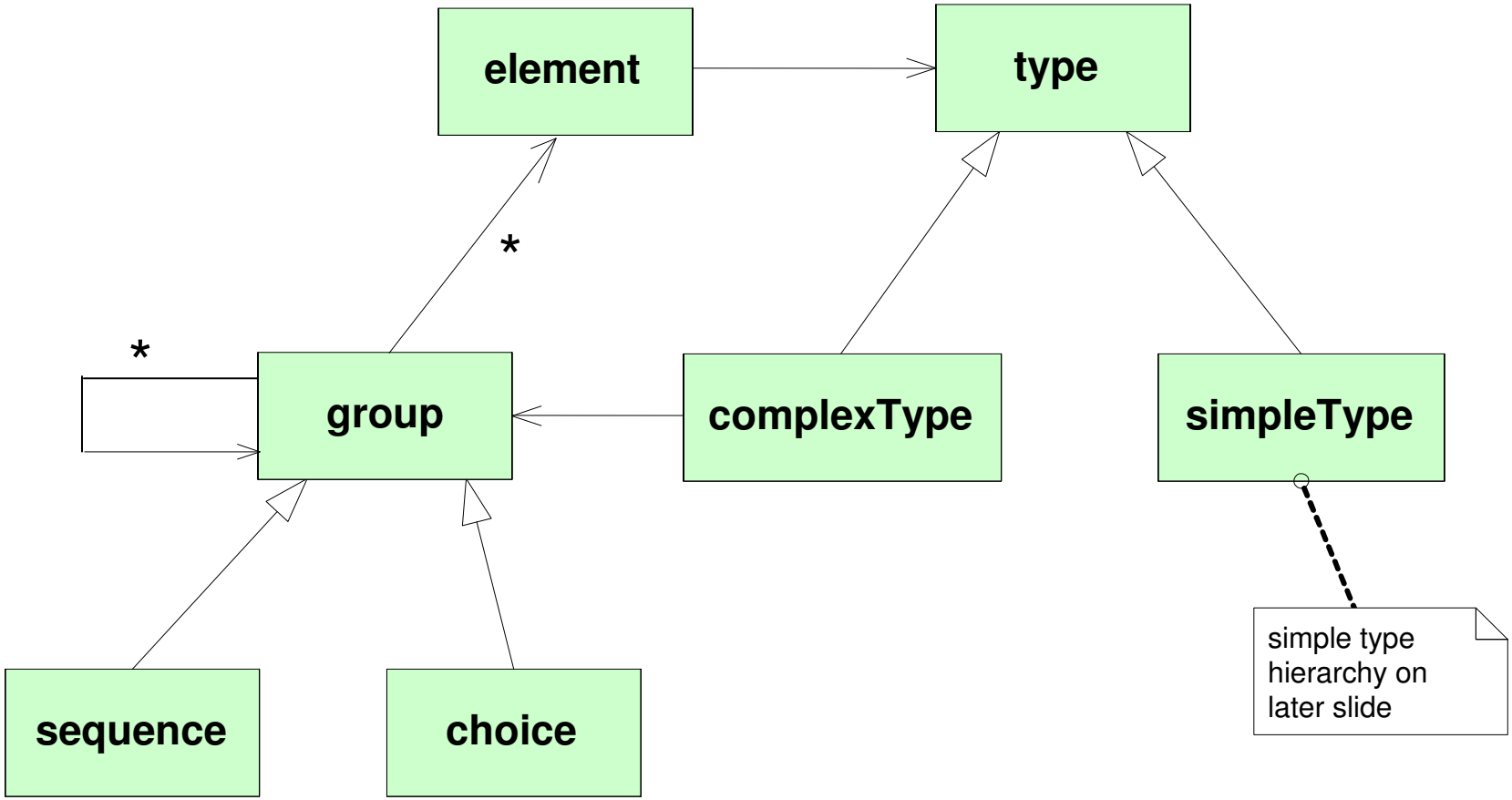
Questions?

Agenda

- Why DFDL?
  - The problem it solves
- What is DFDL?
  - Quick overview for newbies
- DFDL 1.0 status
  - Specification & implementations

- Backup: DFDL Language

DFDL Schema Component Model



DFDL Schema Component Model

OpenGridForum

- Simple Types
  - Represent data values
- Complex Types
  - Represent structures
- Elements
  - Represent named fields
  - Can be data fields or structural fields
  - Can repeat (arrays)
- Sequence groups
  - Structures where the children occur in order
- Choice groups
  - Structures where just one of the children occurs (eg: C union, COBOL redefines)

XML Schema 1.0 - Subset

OpenGridForum

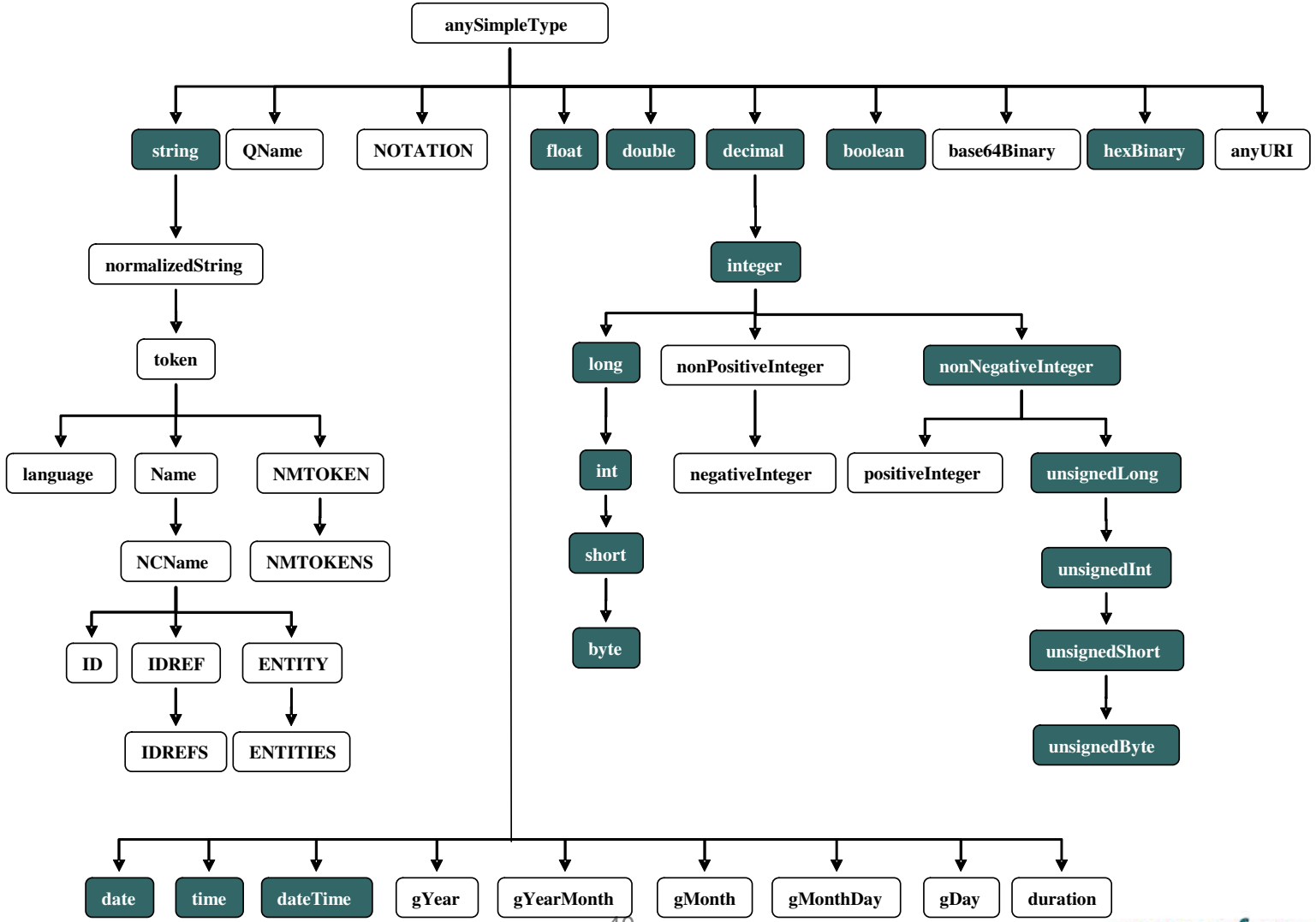
- Namespace management
- xs:import/xs:include file management
- Local and global xs:element declarations
  - Optional dimensionality via maxOccurs and minOccurs
  - Optional default value
  - Optional nillable attribute (nil value support)
- Local and global xs:complexType definitions
- Most built-in xs:simpleTypes – see later slide
- Local and global user-defined xs:simpleTypes by derivation
- Local and global xs:sequence groups (no dimensionality)
- Local and global xs:choice groups (no dimensionality)
- xs:appinfo annotations to carry the DFDL information

XML Schema 1.0 – Not Used

OpenGridForum

- Local and global xs:attribute declarations
- xs:attribute groups
- xs:complexType derivations or with simple/mixed content
- xs:union simple types (except for one specific case)
- xs:list simple types
- Built-in xs:simpleTypes specific to XML or otherwise superfluous
- Dimensionality on xs:sequence and xs:choice
- Identity Constraints
- Substitution Groups
- xs:redefine file management
- Local and global xs:all groups
- xs:any (wildcards)
- Recursion

Supported Simple Types



DFDL Annotations (1/2)

OpenGridForum

Annotation	*Used on Component*	*Purpose*
dfdl:element	xs:element xs:element reference	Contains the DFDL properties of an xs:element and xs:element reference
dfdl:choice	xs:choice	Contains the DFDL properties of an xs:choice.
dfdl:sequence	xs:sequence	Contains the DFDL properties of an xs:sequence.
dfdl:group	xs:group reference	Contains the DFDL properties of an xs:group reference to a group definition containing an xs:sequence or xs:choice.
dfdl:simpleType	xs:simpleType	Contains the DFDL properties of an xs:simpleType
dfdl:format	xs:schema dfdl:defineFormat	Contains a set of DFDL properties that can be used by multiple DFDL schema components. When used directly on xs:schema, the property values act as defaults for all components in the DFDL schema.
dfdl:defineFormat	xs:schema	Defines a reusable data format by associating a name with a set of DFDL properties contained within a child dfdl:format annotation. The name can be referenced from DFDL annotations on multiple DFDL schema components, using dfdl:ref.

DFDL Annotations (2/2)

OpenGridForum

Annotation	*Used on Component*	*Purpose*
dfdl:assert	xs:element, xs:choice xs:sequence, xs:group	Defines a test to be used to ensure the data are well formed. Used only when parsing data.
dfdl:discriminator	xs:element, xs:choice xs:sequence, xs:group	Defines a test to be used when resolving a point of uncertainty such as choice branches or optional elements. Used only when parsing.
dfdl:escapeScheme	dfdl:defineEscape Scheme	Defines a scheme by which quotation marks and escape characters can be specified. This is for use with delimited text formats.
dfdl:defineEscape Scheme	xs:schema	Defines a named, reusable escape scheme. The name can be referenced from DFDL annotations on multiple DFDL schema components.
dfdl:defineVariable	xs:schema	Defines a variable that can be referenced elsewhere. This can be used to communicate a parameter from one part of processing to another part.
dfdl:newVariable Instance	xs:element, xs:choice xs:sequence, xs:group	Creates a new instance of a variable
dfdl:setVariable	xs:element, xs:choice xs:sequence, xs:group	Sets the value of a variable whose declaration is in scope

DFDL Properties

OpenGridForum

- Properties on DFDL annotations may be one or more of the following types:
  1. DFDL string literal
     - Can include DFDL entities for special character(s)
  2. DFDL expression
     - XPath to other parts of the data
     - Can use DFDL variables
  3. Regular expression
  4. Enumeration
  5. Logical value
  6. QName
- Some properties can have alternative values
  - Use a space separated list
- Note: DFDL properties do not have defaults!

Properties – 'Attribute' Form


```
<xs:sequence>
  <xs:annotation>
    <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
      <dfdl:sequence separator=“,” encoding=“ascii”/>
     </xs:appinfo>
  </xs:annotation>
</xs:sequence>

<xs:element name="z" type=“xs:float">
  <xs:annotation>
    <xs:appinfo source="http://www.ogf.org/dfdl/v1.0">
      <dfdl:element representation="text"
          textNumberRep="standard" encoding="ascii"
          lengthKind="delimited" initiator="float=" …/>
  </xs:appinfo>
 </xs:annotation>
</xs:element>
```

Alternative 'Element' Form:

```
<dfdl:property name=“encoding”>ascii</dfdl:property>
```

Properties – 'Short' Form

```
<xs:sequence dfdl:separator=“,” dfdl:encoding=“ascii”/>

<xs:element name="z" type=“xs:float“
            dfdl:representation="text"
            dfdl:textNumberRep="standard"
            dfdl:encoding="ascii"
            dfdl:lengthKind="delimited"
            dfdl:initiator="float=" … />
```

Non-native attribute syntax,
easy for users to write

Properties - Scoping

```
<xs:sequence dfdl:separator=","
             dfdl:encoding="ebcdic-cp-us">
      <xs:element name="s" type="xs:string"/>
      <xs:element name="m" type="myType"/>
</xs:sequence>

<xs:complexType name="myType">
   <xs:sequence>
      …
   </xs:sequence>
</xs:complexType>
```

- Are the elements of the sequence in EBCDIC, or just the separators? (“lexical scoping”)
- Are the contents of type ‘myType' affected by the sequence’s properties or not? (“referential transparency”)

Properties - Scoping Rules

OpenGridForum

- A DFDL property declared in a dfdl:format annotation on the xs:schema itself applies globally to **all** components declared in the schema.
- A local DFDL property declared in a component annotation overrides one obtained from a referenced dfdl:defineFormat annotation via dfdl:ref.
- A local DFDL property declared in a component annotation or obtained via dfdl:ref:
  - applies to that component only
  - does **not** apply to any child components
  - overrides any value from a dfdl:format on the xs:schema
- Algorithm specified for combining properties declared on:
  - xs:simpleType restriction and base xs:simpleType
  - xs:element and referenced xs:simpleType
  - xs:element ref and referenced global xs:element
  - xs:group ref and referenced xs:sequence or xs:choice

Scoping Example

OpenGridForum

aaa,bbb,ccc

What encodings ?

```
<xs:schema>
  <xs:annotation>
    <xs:appinfo source="http://www.ogf.org/dfdl/" >
       <dfdl:format encoding="ascii" representation="text"
                    lengthKind="delimited" … />
    </xs:appinfo>
  </xs:annotation>

  <xs:annotation>
    <xs:appinfo source="http://www.ogf.org/dfdl/" >
       <dfdl:defineFormat name="myFormat" />
          <dfdl:format encoding="utf-8" … />
       </dfdl:defineFormat>
    </xs:appinfo>
  </xs:annotation>

  <xs:complexType>
    <xs:sequence dfdl:separator="," dfdl:ref="myFormat" >
       <xs:element name="a" type="xs:string" />
       <xs:element name="b" type="xs:string" dfdl:encoding="iso-8859-1" />
       <xs:element name="c" type="xs:string" dfdl:ref="myFormat"
                                             dfdl:encoding="iso-8859-1" />
     </xs:sequence>
   </xs:complexType>
</xs:schema>
```

Answers:
separator: utf-8
aaa: ascii
bbb: iso-8859-1
ccc: iso-8859-1

DFDL Data Syntax Grammar

OpenGridForum

- Data in a format describable using a DFDL schema obeys the DFDL syntax grammar
- Data is divided into:
  - Content – used to compute logical values
  - Framing – delimiters, alignment, etc
- The specification organises DFDL properties according to the grammar productions.

Grammar - Productions

OpenGridForum

Productions
Document = Element
Element = SimpleElement | ComplexElement SimpleElement = ElementLeftFraming SimpleContent RightFraming ComplexElement = ElementLeftFraming ComplexContent RightFraming LeftElementFraming = LeftFraming PrefixLength PrefixLength = SimpleContent
LeftFraming = LeadingAlignment ***Initiator*** RightFraming = ***Terminator*** TrailingAlignment LeadingAlignment = ***LeadingSkip AlignmentFill*** TrailingAlignment = ***TrailingSkip***
ComplexContent = Sequence | Choice SimpleContent = ***LeftPadding SimpleRepresentation RightPadOrFill***
Sequence = LeftFraming SequenceContent RightFraming SequenceContent = [***PrefixSeparator*** SequenceItem [***Separator*** SequenceItem]* ***PostfixSeparator***] ***FinalUnusedRegion*** SequenceItem = Element | Array | ComplexContent
Choice = LeftFraming ChoiceContent RightFraming ChoiceContent = [Element | Array | ComplexContent] ***FinalUnusedRegion***
Array = [Element [***Separator*** Element]* [***Separator*** StopValue]] StopValue = SimpleElement

Properties - Groupings

- Properties on DFDL annotations are organized according to the grammar productions:

1. Common to Content & Framing
   - encoding, byte order
2. Framing
   - alignment, initiator, terminator
   - length extraction ***(dfdl:lengthKind)***
3. Simple types
   - Number, Calendar, String, Opaque, Boolean
   - text, binary ***(dfdl:representation)***
   - further representation properties
   - nil value, default value
4. Sequences
   - separator, ordering, floating elements
5. Choices
6. Arrays
7. Calculated values

Controls how data values are extracted

Controls how data values are interpreted

Nils & Defaults

OpenGridForum

- A *nil value* is used to handle 'out of band' values
- A *default value* is used when a required element is missing

- **Nil values**
  - Used on parsing and unparsing
  - Simple elements only, xs:nillable must be true
  - Nil value can be logical value, literal value or literal character
  - Specified by dfdl:nilKind & dfdl:nilValue properties

- **Default values**
  - Used on parsing and unparsing
  - For simple elements, xs:default or xs:fixed gives the value
    - Can also specify that nil value is to be used as default value
  - A complex element has a 'default' if all its required children have
    - Allows 'sparse' infosets to be supplied when unparsing
  - Definition of 'required':
    - Scalar (minOccurs=maxOccurs=1)
    - Array of fixed occurences
    - Array of variable occurrences and index <= minOccurs)

Calculated Values

OpenGridForum

- Parsing
  - Sometimes you want to insert an item into the infoset that is not directly in the data
  - Example: create a simple infoset item from several other data values
  - Use dfdl:inputValueCalc="{ expr }"
  - Often used alongside 'hidden' elements
- Unparsing
  - Sometimes you want to set a data value where there is no corresponding infoset item
  - Example: a value that contains the length of another data value
  - Use dfdl:outputValueCalc="{ expr }"

Points of Uncertainty

OpenGridForum

- A point of uncertainty occurs in the data stream when there is more than one schema component that might occur at that point.
- A point of uncertainty is caused when one of the following constructs is used in a DFDL schema:
  - xs:choice
  - Unordered xs:sequence (dfdl:sequenceKind="unordered")
  - xs:element which is optional (minOccurs="0", maxOccurs="1)"
  - xs:element is an array with a variable number of occurrences (minOccurs <> maxOccurs & maxOccurs>"1")
  - xs:sequence containing one or more "floating" elements

Resolving Points of Uncertainty

OpenGridForum

- A DFDL parser is a recursive-descent parser with look ahead used to resolve points of uncertainty.
- The parser must speculatively attempt to parse data until a component is either ‘known to exist’ or ‘known not to exist’.
- Until that applies, the occurrence of a processing error causes the parser to suppress the error, back track and make another attempt.
- A component is ‘known to exist’ if either:
  - All the syntax and content of the component are successfully parsed and any dfdl:assert on the component evaluates to true.
  - A dfdl:discriminator on the component evaluates to true
  - The parent xs:sequence or xs:choice has dfdl:initiatedContent “Yes” and the initiator for the component is found
- Each point of uncertainty construct has its own rules.

DFDL Expressions

OpenGridForum

- Can use a DFDL expression:
  - When a property value needs to be set dynamically at parse time from the contents of one or more elements of the data (eg, dfdl:separator, dfdl:length)
  - In a dfdl:assert annotation
  - In a dfdl:discriminator annotation to resolve uncertainty
  - In a dfdl:inputValueCalc property to derive the value of an element in the infoset that doesn't exist in the data
  - In a dfdl:outputValueCalc property to compute the value of an element on output
  - As the value in a dfdl:setVariable annotation or the defaultValue in a dfdl:defineVariable
- Eg: dfdl:length = "{ $mylen + 1 }"

Expression Language

- Subset of XPath 2.0
  - Only *if* and *path* expression types
  - Only *child*, *parent* and *self* axes
  - Predicates but only to index arrays
  - Subset of functions and operators
- Plus some extensions
  - Additional constructor functions
  - DFDL functions for representation lengths, property values, bit manipulation
- Variables
  - Use values of variables set up by dfdl:defineVariable, dfdl:newVariableInstance and dfdl:setVariable annotations
- Operates on *augmented* infoset so that hidden elements can be accessed

Expressions Example (1/2)

```
<xs:sequence>
  <xs:sequence
     dfdl:hiddenGroupref="hiddenpDate" />


  <xs:element name="d" type="xs:date">
    <xs:annotation><xs:appinfo source="http://www.ogf.org/dfdl/">
      <dfdl:element>
        <dfdl:property name="inputValueCalc">
         { fn:date(fn:concat(if (../pdate/yy gt 50) then "19" else "20",
                             if (../pdate/yy gt 9)
                                then fn:string(../pdate/yy)
                                else fn:concat("0", fn:string(../pdate/yy)),
                             "-",
                             fn:string(../pdate/mm),
                             "-",
                             fn:string(../pdate/dd)))
         }
        </dfdl:property>
      </dfdl:element>
    </xs:appinfo></xs:annotation>
  </xs:element>
</xs:sequence>
```

dfdl:hiddenGroupRef wraps complex "pDate" element that we don't want in the infoset

dfdl:inputValueCalc expression creates an xs:date from hidden "pDate"

Expressions Example (2/2)

```
<xs:group name="hiddenpDate">
   <xs:sequence>
     <xs:element name="pdate">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="mm" type="xs:byte"
               dfdl:lengthKind="explicit" dfdl:length="1"
               dfdl:representation="binary" dfdl:binaryNumberRep="binary"
               dfdl:outputValueCalc="{ fn:month-from-date(../d) }" />
            <xs:element name="dd" type="xs:byte"
               dfdl:lengthKind="explicit" dfdl:length="1"
               dfdl:representation="binary" dfdl:binaryNumberRep="binary"
               dfdl:outputValueCalc="{ fn:day-from-date(../d) }" />
            <xs:element name="yy" type="xs:byte"
               dfdl:lengthKind="explicit" dfdl:length="1"
               dfdl:representation="binary" dfdl:binaryNumberRep="binary"
               dfdl:outputValueCalc="{ fn:year-from-date(../d) idivmod 100 }"/>
          </xs:sequence>
        </xs:complexType>
     </xs:element>
  </xs:sequence>
</xs:group>
```

Complex “pDate” element models a date as 3 x 1 byte integers

dfdl:outputValueCalc expression gets the value from the “d” xs:date element

Questions?