Kill the Data Format Problem

with Apache Daffodil™ (Incubating)

Mike Beckerle, Data Archeologist, Owl Cyber Defense
mbeckerle at owlcyberdefense.com or at apache.org

ApacheCon @Home Sep 29 - Oct 1, 2020

Chat (or Hoot) in the ApacheCon Slack #incubator channel

Got EDIFACT Data?


```
UNA:+.?*'
UNB+UNOC:4+5790000274017:14+5708601000836:14+990420:1137+17++INVOIC++++1'
UNH+30+INVOIC:D:03B:UN'
BGM+380+539602'
DTM+137:19990420:102'
RFF+CO:01671727'
NAD+BY+5708601000836::9'
RFF+VA:UK37499919'
NAD+SU++IBM UK'
RFF+VA:UK19430839'
RFF+ADE:00000767'
NAD+DP+++MyCompany+MyStreet+MyTown++1234+UK'
CUX+2:GBP:9'
LIN+1++V0370246:IN'
IMD'F'''::Collectors edition of The Hobbit with Tolkien?'s original colours on slee
```

Got bit-packed binary data ?

- Bytes are

  09 20 42 F0 0D B8 DD

- Fields are decribed as:

Message Number	XXXXXX00 00001xxx
FPI for Message Subtype	XXXXX0xx
FPI for File Name	XXXX0xxx
FPI for Message Size	XXX0XXXX
Operation Indicator	X01XXXXX
Retransmit Indictor	0XXXXXXX
Message Precendence Codes	XXXXX010
Security Classification	XXX00XXX
FPI for C/R marking	XX0XXXXX

Got NACHA Data?


```
101 121000248 1210002480608080107A094101WFB-W EDI CUST. DATA   WFB-E ACH SPINAT DATA
5200APD TX/FINCL SVC323413684          9666666606CCDAPD - TAX 06080206080221410210000240306491
627107005432000040011914770   0542151200614007046488KD8BRODY LABORATORIES INCFS0021000024030840
820000000100107005430005421512000000000000009666666606                    021000024030649
5200ARAR                               9000290001CCD PAYMENT        0608072191021000027294149
622051001412000049450370590   0000403676239412800 VIA LICENSING CORP      0021000027294283
820000000100051001410000000000000000040367690002900010                    021000027294149
5200ACME   WORLDWIDEDIRECT DEPOSIT      9954245682CCDA/P             0608022141122000030000219
622107005432000040011914770   0000050000100001426110080 100815047371712006      0122000036548030
622107005432000040011914770   0000147500100001426110080 100815047375772006      0122000036548031
820000000200214010860000000000000000019750099542456820                    122000030000219
5200BEST BANK NA    5046001042958      9560900031CCDEDI PAYMNT      0511181231021101100000014
622051001414999527563877100000375005040589670      XXXXXXXX BRANCH INC     1021101100001681
705RMR*OI*0140611**-1170.49*25*1170.36*                                    00010001681
622061000227999012461728300235195455040589680     XXXXXXXXXXXXX SERVICE   1021101100001682
705RMR*ADG*504058968                                                       00010001682
```

Got ISO8583 Data?

```
1111FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF1912345678901234
5678912345612345678901212345678901212345678901212312
3595912345678123456781234567812345699123112000099129
9129912311231123112341231231 23A1B2C3D4E5F61231231234
1234199123112312345678901234567890123408C00015001112
3456789011112345678901281234567890123456789012345678
26;1111111111111111111=1215=?1062;11222222222222222222
2=12311234123412341211234561121212123 41?1A1B2C3D4E5F
6A1B2C3123123A1A#A1A#A1#A1#A1#A1#A1#15A1#A1#A1#A1#A1
#15A1#A1#A1#A1#A1#35%A1111111111111111111^JOHNDOE^1215
^?015A1#A1#A1#A1#A1#015A1#A1#A1#A1#A1#015A1#A1#A1#A1
#A1#ABCABCABC0800812345678008011123011001100110011 00
110011
```

Data Format Description Language

DFDL →DaFfoDiL

- DFDL is a way of describing data formats
- It is NOT a data format itself!

- Open Standard from the Open Grid Forum (OGF)
- DFDL Specification - Expecting Final v1.0 in 1H 2021.
  - 2 other DFDL Implementations (IBM, ESA)

- DFDL standard = union of capabilities across many marketplace data integration products/tools

Use Daffodil: NACHA as JSON Please...


```
{ "ACHFile":
{ "FileHeaderRecord":
{ "RecordTypeCode": "1",
"PriorityCode": "01",
"ImmediateDestination": " 123456789",
"ImmediateOrigin": " 987654321",
"FileCreationDate": "071030",
"FileCreationTime": "1634",
"FileIdModifier": "A",
"RecordSize": "094",
"BlockingFactor": "10",
"FormatCode": "1",
"ImmediateDestinationName": "TEST Destination ", "ImmediateOriginName": "TEST
Origination ", "ReferenceCode": " " },
"Batch": [ {
"BatchHeaderRecord": {
"RecordTypeCode": "5",
"ServiceClassCode": "200",
"CompanyName": "VIA LICENSING CO",
```


Prefer my NACHA as XML Please....


```
<ACHFile xmlns="ach:2013">
<FileHeaderRecord>
<RecordTypeCode>1</RecordTypeCode>
 <PriorityCode>01</PriorityCode>
<ImmediateDestination> 123456789</ImmediateDestination>
<ImmediateOrigin> 987654321</ImmediateOrigin>
<FileCreationDate>071030</FileCreationDate>
<FileCreationTime>1634</FileCreationTime>
<FileIdModifier>A</FileIdModifier>
 <RecordSize>094</RecordSize>
<BlockingFactor>10</BlockingFactor>
 <FormatCode>1</FormatCode>
<ImmediateDestinationName>TEST Destination </ImmediateDestinationName>
<ImmediateOriginName>TEST Origination </ImmediateOriginName>
<ReferenceCode> </ReferenceCode>
</FileHeaderRecord>
```

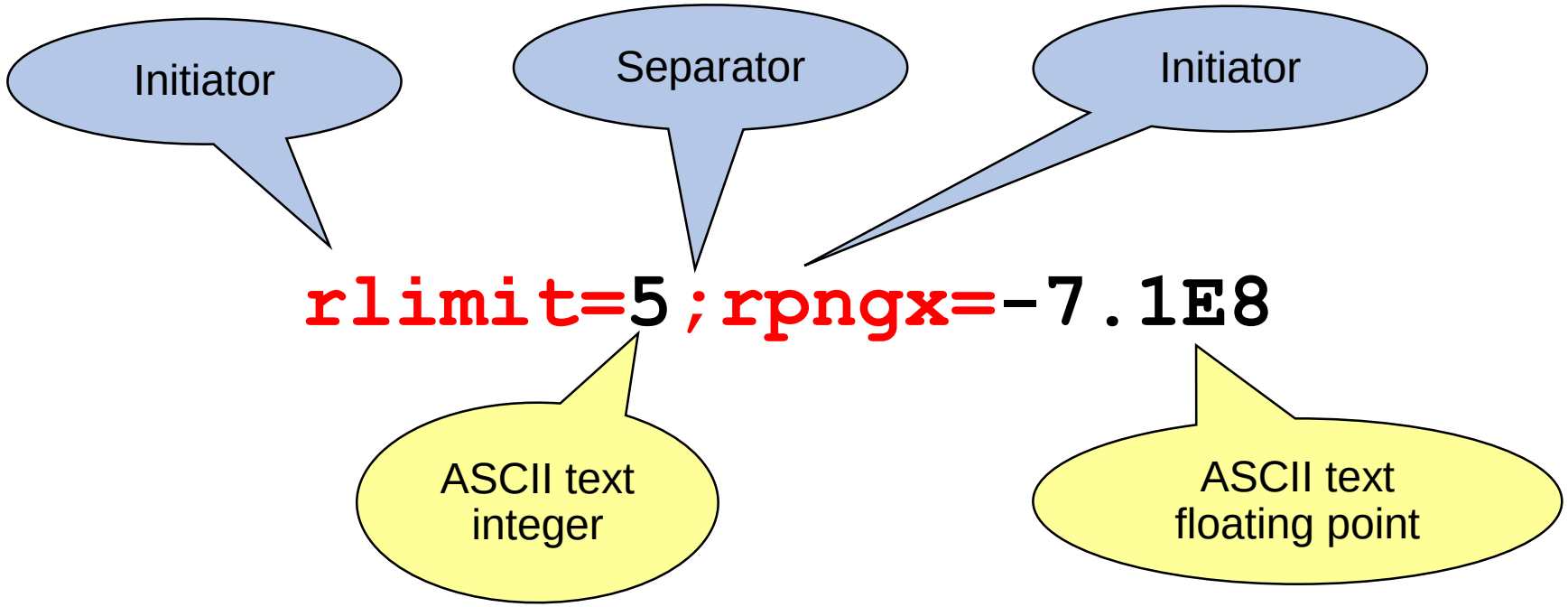
Example DFDL Schema

DFDL in 4 slides!

Example – Delimited Text Data

```
rlimit=5;rpngx=-7.1E8
```

Example – Delimited Text Data



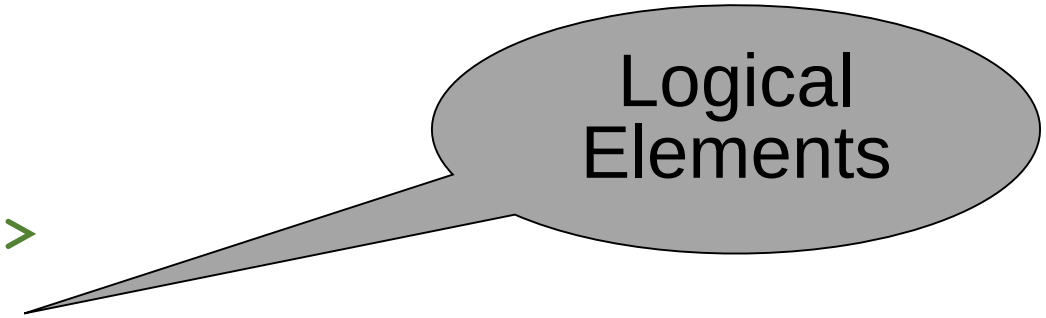
Red is *framing*. Black is *content*.
Separators, initiators (aka tags) are *delimiters* (which are framing).

DFDL Schema uses XML Schema

```
<xs:complexType name="rValues">
  <xs:sequence>
    <xs:element name="rlimit" type="xs:int"/>

    <xs:element name="rpngx" type="xs:float"/>

  </xs:sequence>
</xs:complexType>
```



DFDL schema

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

<xs:complexType name="rValues">
  <xs:sequence dfdl:separator=";" ... >
    <xs:element name="rLimit" type="xs:int"
                  dfdl:initiator="rLimit=" ... />
    <xs:element name="rpngx" type="xs:float"
                  dfdl:initiator="rpngx=" ... />
  </xs:sequence>
</xs:complexType>
```

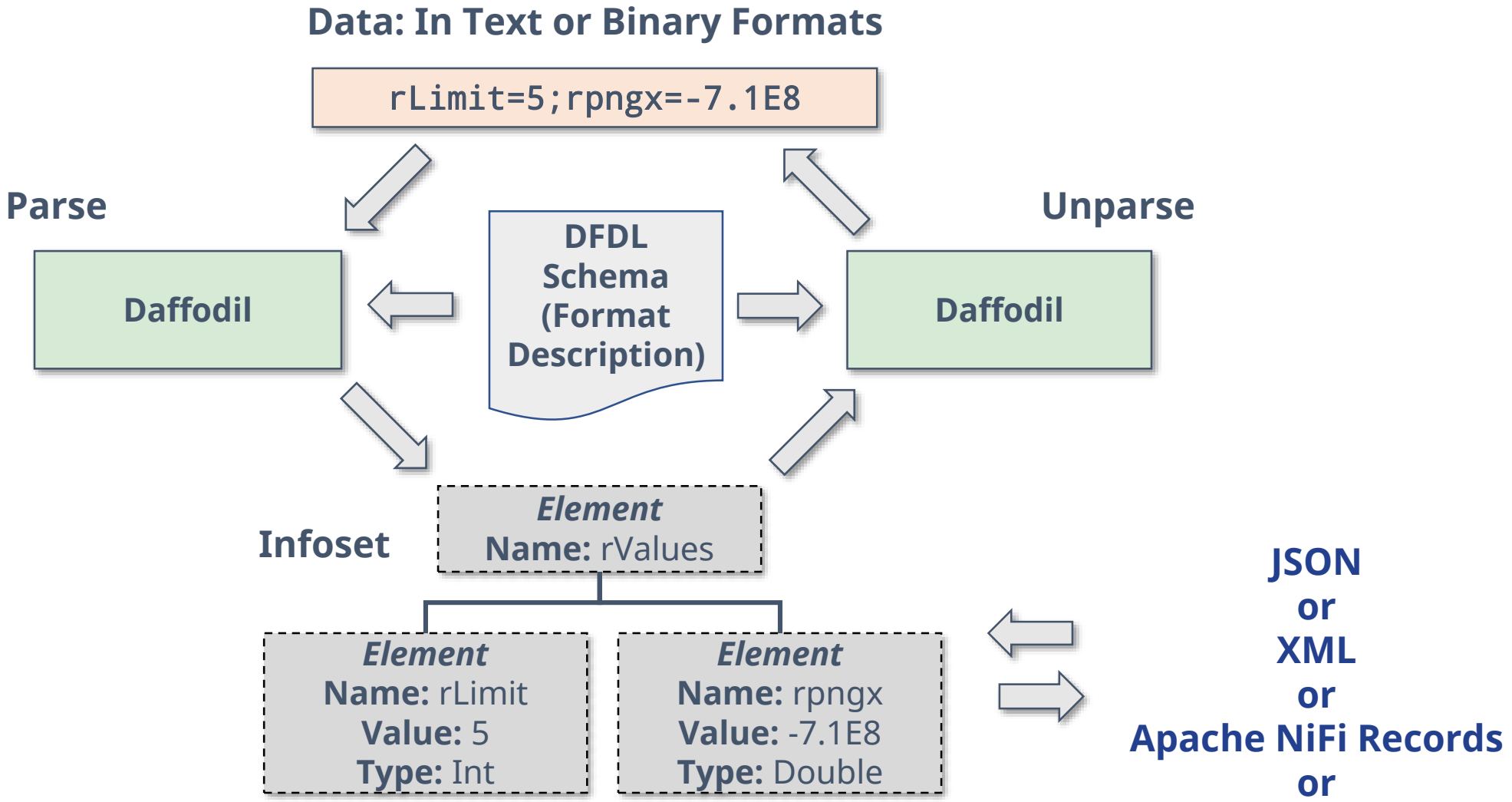
;

rLimit=5

rpngx=-7.1E8

DFDL properties

DFDL Parsing and Unparsing



DFDL Schemas

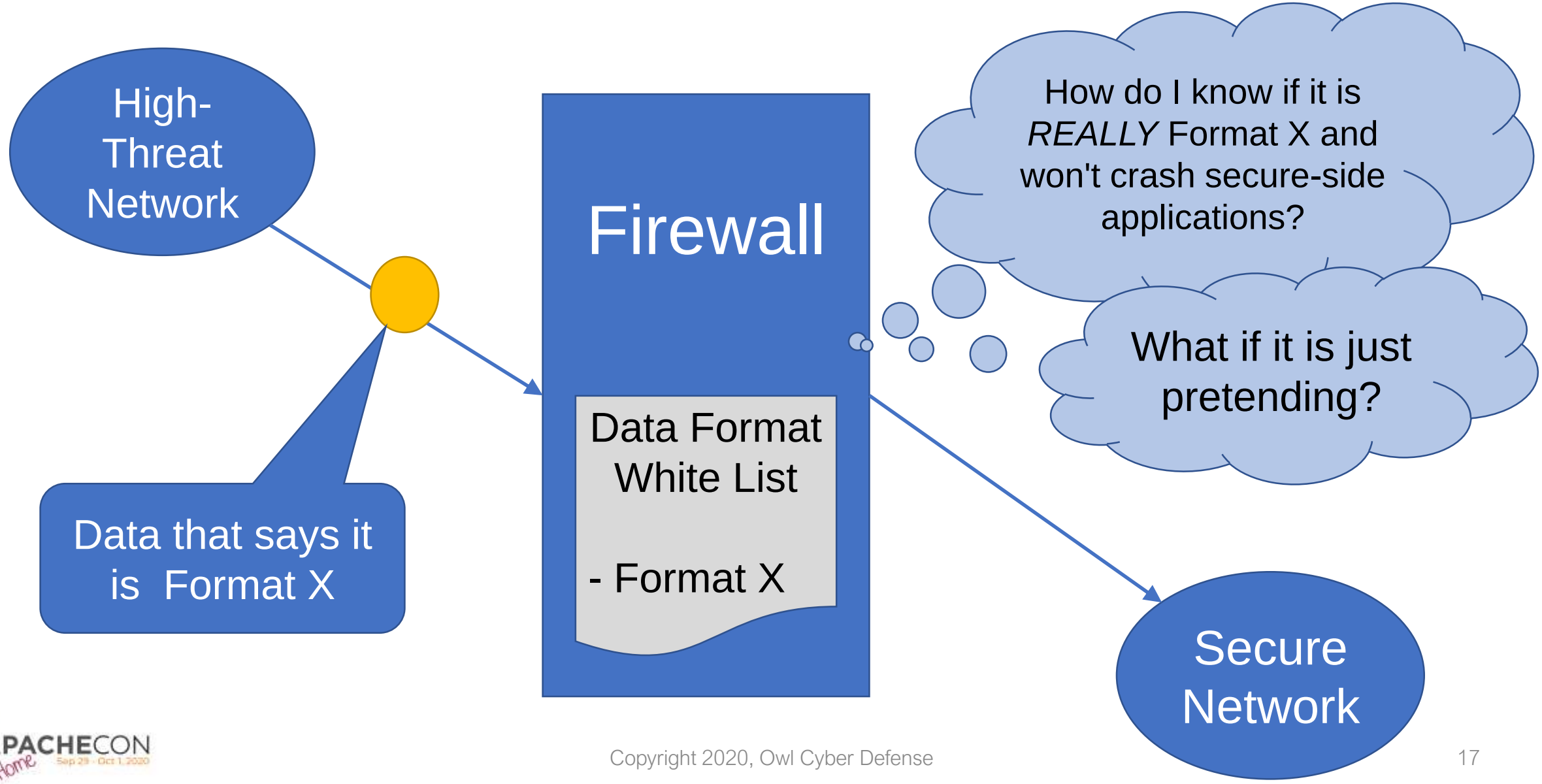
Public (most on github)	MIL-STD-2045 PCAP NITF PNG JPEG NACHA VCard QuasiXML Geonames CSV	EDIFACT IBM4690-TLOG ISO8583 BMP GIF Praat TextGrid ARINC429-PoC IPFIX Syslog	iCalendar IMF SHP (shape file) KNXNet/IP(indust. control) Siemens S7 (indust. control) Asterix (Cat 034, 048) MagVar AFTN Flight Plan RASTER (RPF) ICD-GPS-240
FOUO / CUI	VMF VMF_S2S unit-normalizing (Rev A) USMTF ATO (MIL-STD-6040) LINK16 (NATO STANAG 5516) LINK16 (MIL-STD-6016F subset) A-GNOSC REMEDY ARMY DRRS USCG UCOP CEF-R1965 GMTIF (STANAG 4607)		SOTF JICD NACT JREAP-C DISV6 SIMPLE (STANAG 5602 Ed 3) P8 JANAP-128
Commercial License $$$	SWIFT-MT (IBM) HIPAA-5010 (IBM) HL7-2.7 (IBM)	USMTF ATO, ACO, etc. (Owl) LINK16 (MIL-STD-6016 E, F, G) (Owl) VMF (MIL-STD-6017 A, B, C, D) (Owl)	

Kill the Data Format Problem

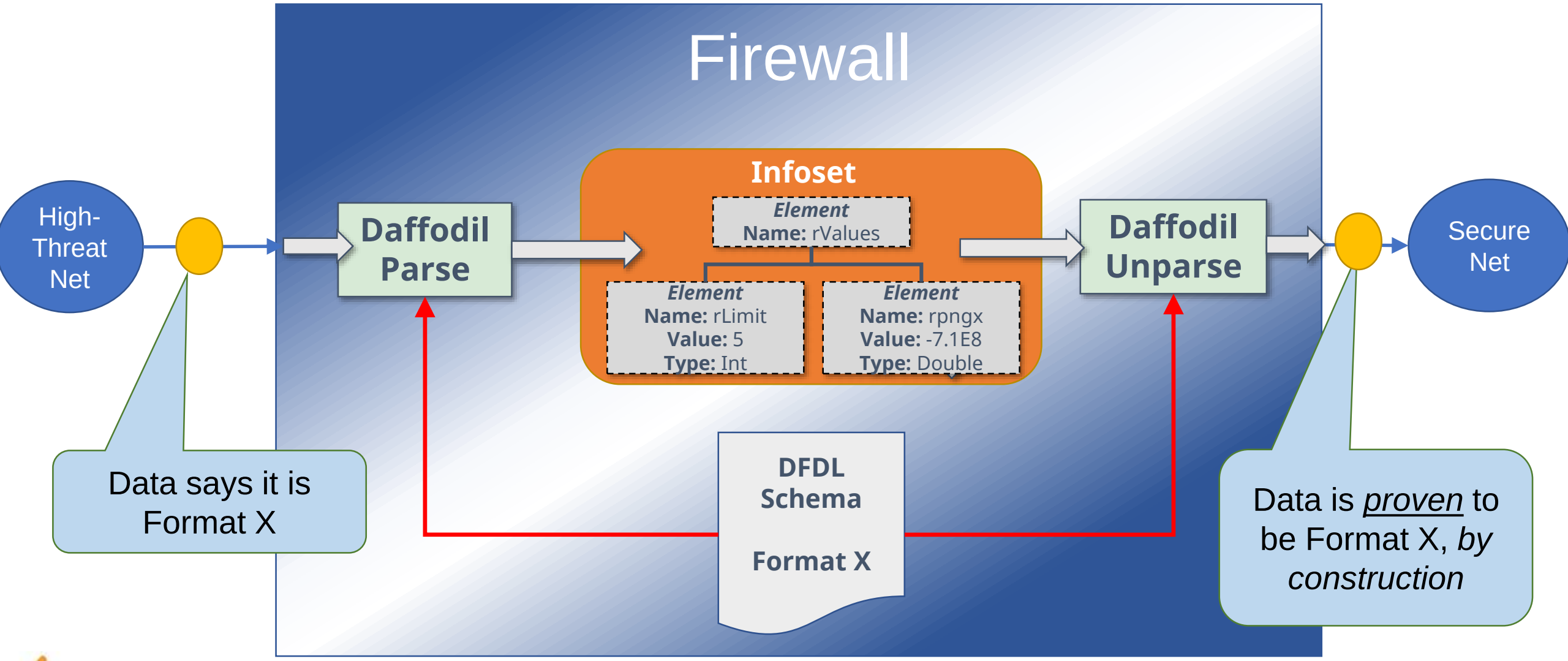
To really kill this problem you need....

- An Open Standard DFDL Language
  - Multiple implementations that interoperate
    - Commercial & Open Source
  - Long-term sponsors with compelling use case
    - IBM – has their own DFDL implementations
    - US DoD, Canada DND - *Cybersecurity*
- A High-Quality Open Source Library Implementation
  - With a supporting community of developers
  - With available commercial support (Owl)

Cyber-Security Use Case: Bad Data DOS



Cyber-Security Use Case

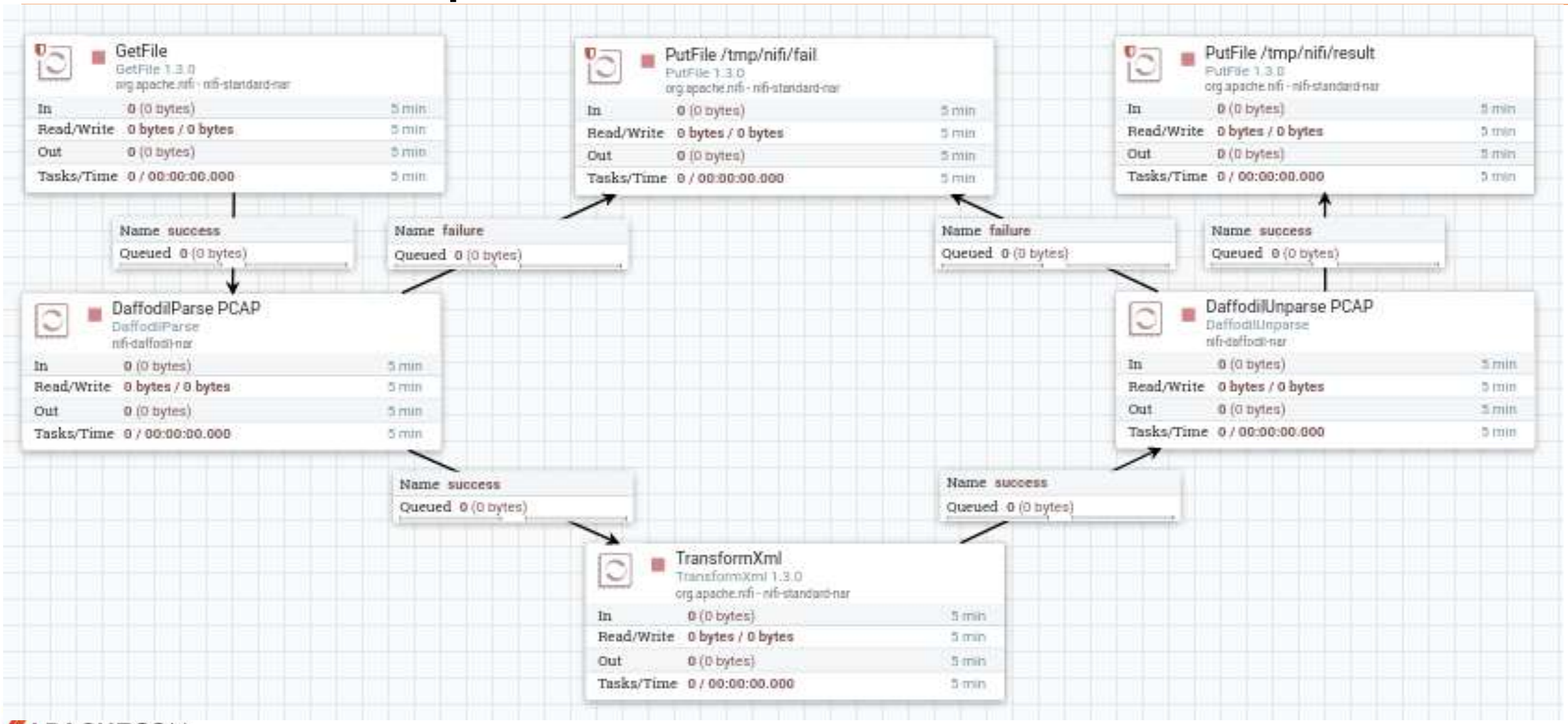


Daffodil Integrations

- Apache NiFi (Native)
- Apache Spark (via XML)
- XProc - Calabash
- SoftwareAG™ Integration Server (aka WebMethods™)(via XML)
- Owl Cyber Defense products
- Other companies products

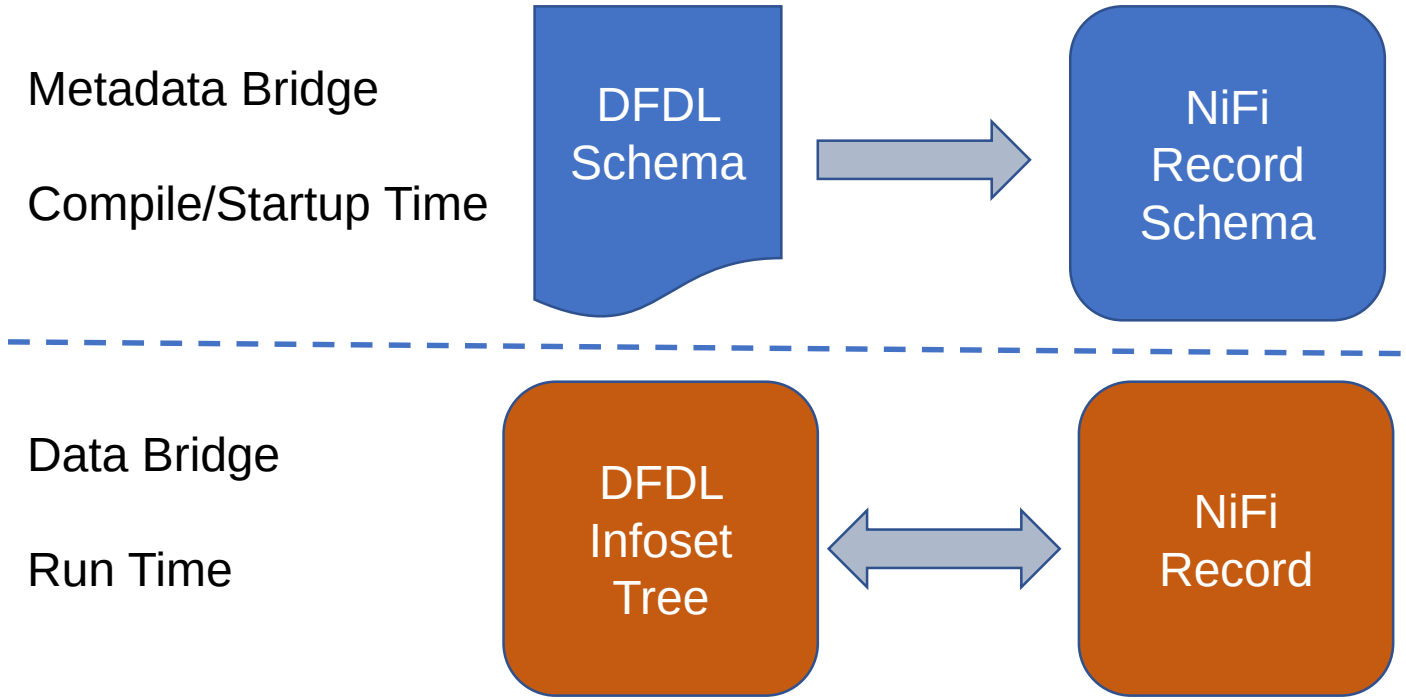
- Important Potential for *Native* Integrations
  - Tika, Spark, Flink, Beam, Hadoop, Storm, Drill, ...

Daffodil + Apache NiFi



Daffodil + Apache NiFi

- Direct Native Data and Metadata Bridge
- Bypasses all XML/JSON overhead



A similar native integration can be done for any of the big data frameworks

Daffodil - Incubator Status

- Podling since Sept 2017
- 7 Apache Releases
- Written in *Scala* - runs on JVM - has Java API

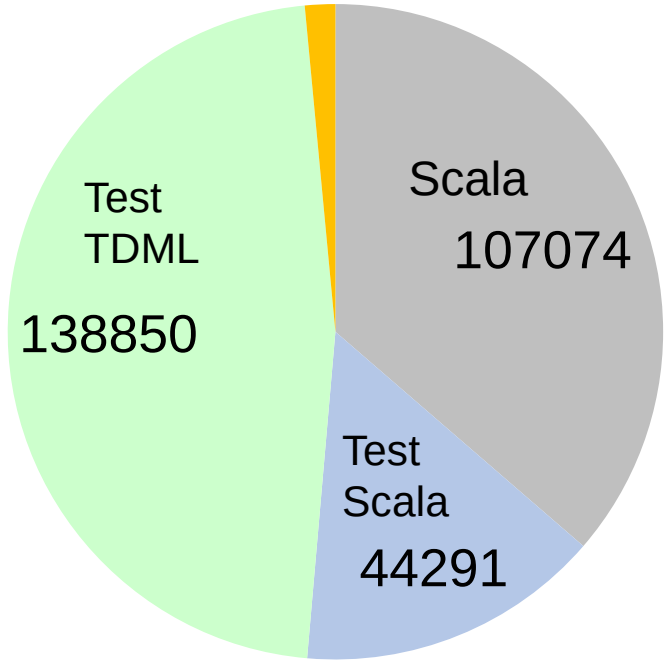
- Status: Community Building
  - Ready for new developers
    - 60 "Beginner" JIRA Tickets
  - Interesting new areas for contribution

Cool Daffodil Ideas ... Already Underway

- Extend Daffodil with SAX-style event streaming
  - Handle data much larger than fits in memory
- Tight Integration with Big Data Frameworks
  - Daffodil directly constructs framework-native data representation with tight metadata coupling
  - Removes the XML/JSON conversion overheads
- Ultra-fast small-footprint backend for Daffodil
  - Generate C code to parse/unparse
  - Generate FPGA logic for wire-speed parse/unparse
- Data-Format Debugger/IDE
  - Graphical display shows data and parsed tree and schema interactively

Developers Wanted

Lines of Code = 295K



- Daffodil is big and complex
  - Full-blown compiler for DFDL schemas
  - Efficient low-level runtime for parse/unparse
- Know or Learn Scala
- Know or Learn XML Schema

- We need developers ....
  - *undaunted* by these challenges
  - motivated by the desire to *kill the data format problem* once and for all.

Questions?

DFDL Specification: http://ogf.org/dfdl

DFDL Schemas: https://github.com/DFDLSchemas

Daffodil Open Source: https://daffodil.apache.org

END

Extra slides after this are available in case of need for discussion.

Why is a *standard* DFDL Needed?

There are *hundreds* of ad-hoc data format description systems

Every Enterprise Software Company

- IBM (10+)
- Oracle(10+)
- SAP(10+)
- Microsoft
- SAS
- Informatica
- SyncSort
- AbInitio
- Pervasive
- Qlik/Expressor
- Pentaho
- .... Dozens more

Every kind of software that takes in data:

- data directed routing (msg brokers)
- database
- data analysis and/or data mining
- data cleansing
- master data management
- application integration
- visualization

All these data format descriptions are:

- *proprietary*
- *ad-hoc*
- *incompatible*

Even within products of the same company!

Why a standard DFDL is Needed?

- Hundreds of data format description systems… means that:
- Vendor investment is spread too thin
  - Tools for creating data formats are inadequate
  - No product is comprehensive enough
  - Some products aren't fast enough
- Customers get locked in to proprietary solution
- High training cost
- Inflexible packaging
  - not libraries - must embed some product in your application data flow

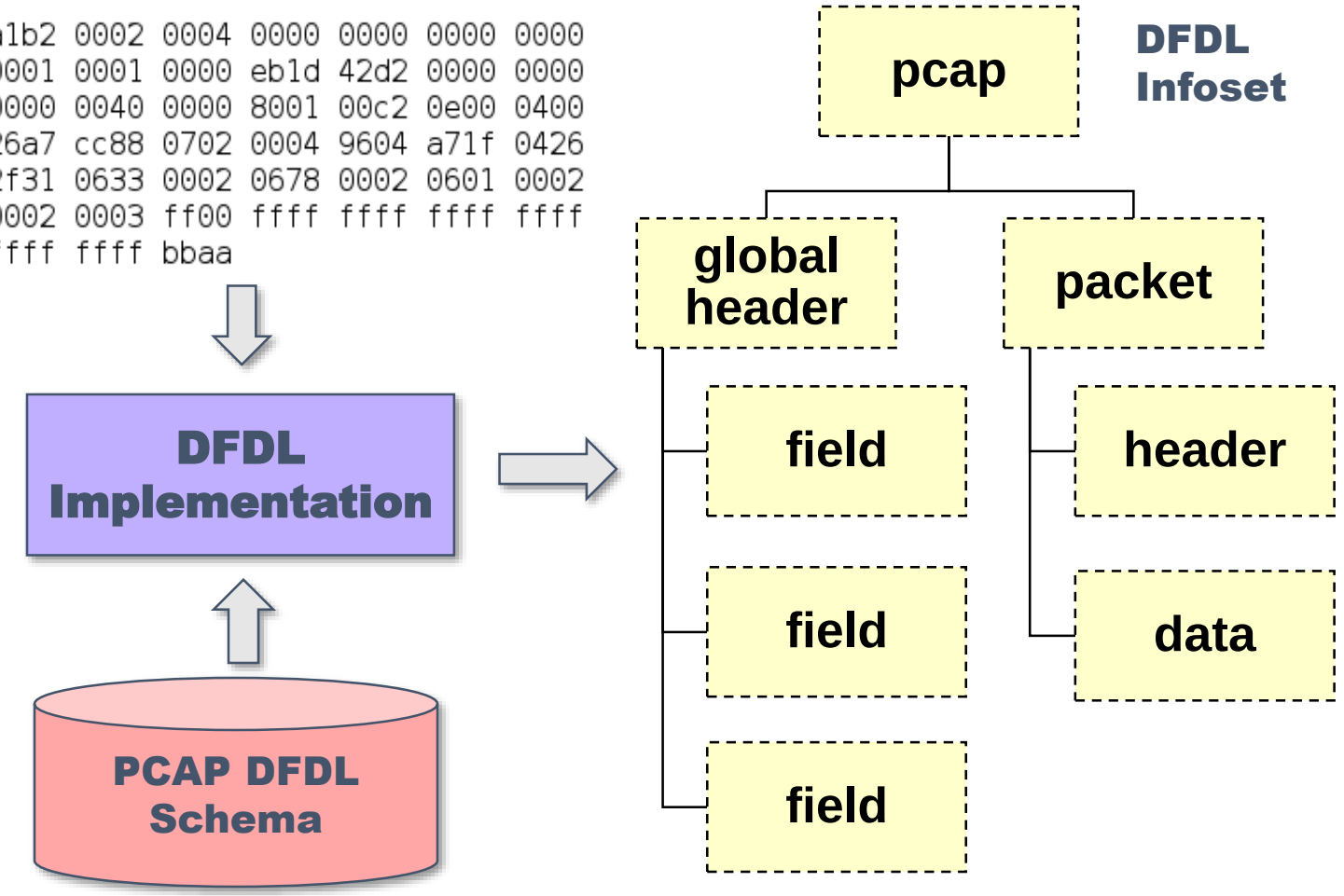
Example: DFDL Advances State of the Art with Computed Elements and Unparsing

Packet Capture (PCAP) Format

Example - Binary Data

Data

```
c3d4 a1b2 0002 0004 0000 0000 0000 0000
9000 0001 0001 0000 eb1d 42d2 0000 0000
0040 0000 0040 0000 8001 00c2 0e00 0400
1f96 26a7 cc88 0702 0004 9604 a71f 0426
0504 2f31 0633 0002 0678 0002 0601 0002
0602 0002 0003 ff00 ffff ffff ffff ffff
ffff ffff ffff bbaa
```



Example - PCAP DFDL Schema

```
<xs:element name="PacketHeader">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="Seconds" type="pcap:uint32"/>
         <xs:element name="USeconds" type="pcap:uint32"/>
         <xs:element name="InclLen" type="pcap:uint32"
            ...
         />
         <xs:element name="OrigLen" type="pcap:uint32"
            ...
          />
      </xs:sequence>
   </xs:complexType>
</xs:element>
...
<xs:element ref="pcap:LinkLayer"
   dfdl:lengthUnits="bytes" dfdl:lengthKind="explicit"
   dfdl:length="{ ../PacketHeader/InclLen }"/>
```

Example - PCAP

```
<xs:element name="PacketHeader">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="Seconds" type="pcap:uint32"/>
         <xs:element name="USeconds" type="pcap:uint32"/>
         <xs:element name="InclLen" type="pcap:uint32"
           dfdl:outputValueCalc="{
            if (dfdl:valueLength(
                     ../../pcap:LinkLayer/pcap:Ethernet,
                     'bytes') le 60) then 60
            else
               dfdl:valueLength(
                     ../../pcap:LinkLayer/pcap:Ethernet,
                     'bytes') }"
         />
         <xs:element name="OrigLen" type="pcap:uint32"
            dfdl:outputValueCalc="{ ../InclLen }"/>
      </xs:sequence>
   </xs:complexType>
</xs:element>
...
<xs:element ref="pcap:LinkLayer"
   dfdl:lengthUnits="bytes" dfdl:lengthKind="explicit"
   dfdl:length="{ ../PacketHeader/InclLen }"/>
```

Example - PCAP