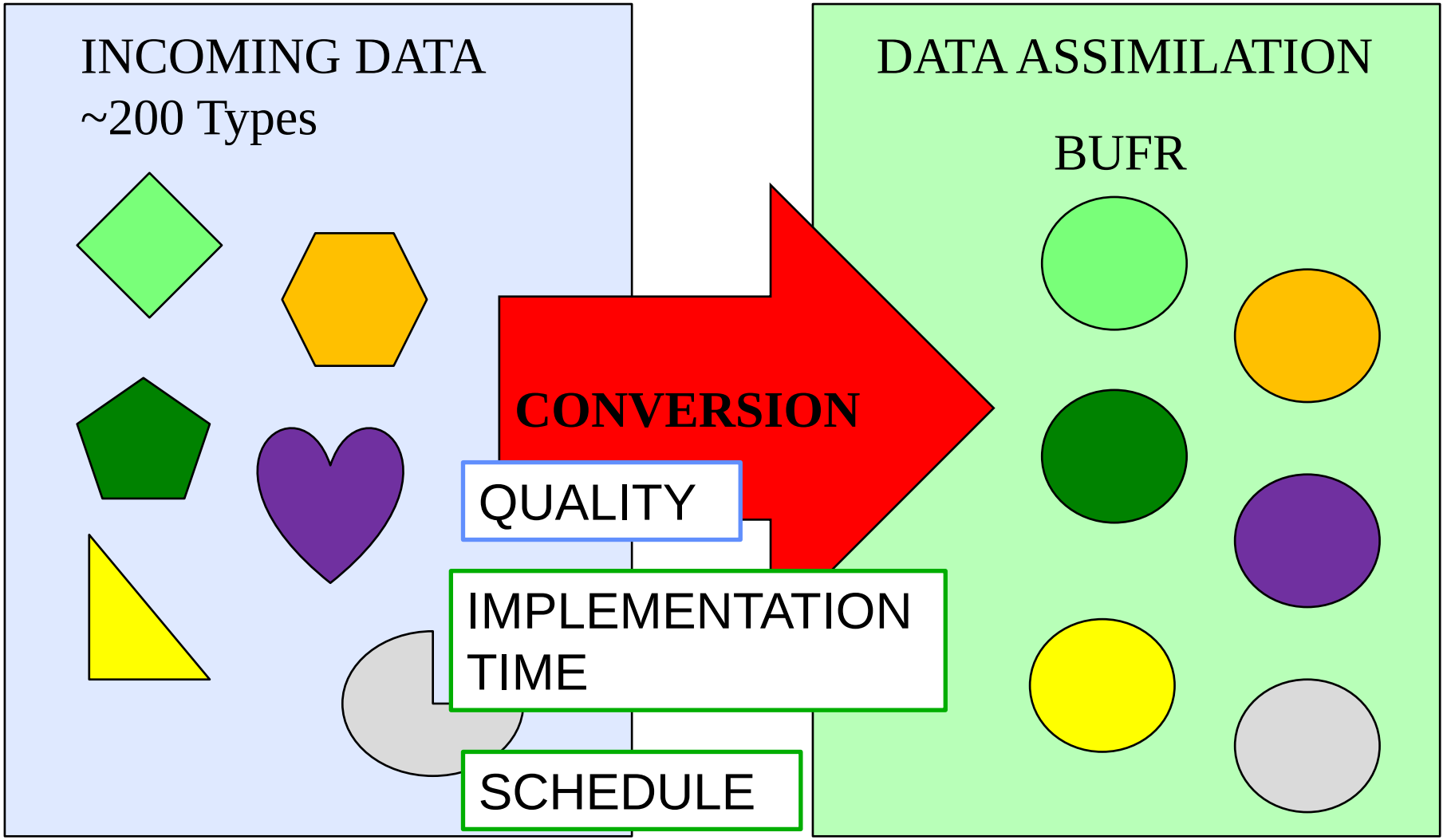
Conversion of METAR to AvXML (IWXXM) using ecCodes and PyXB

Enrico Fucile

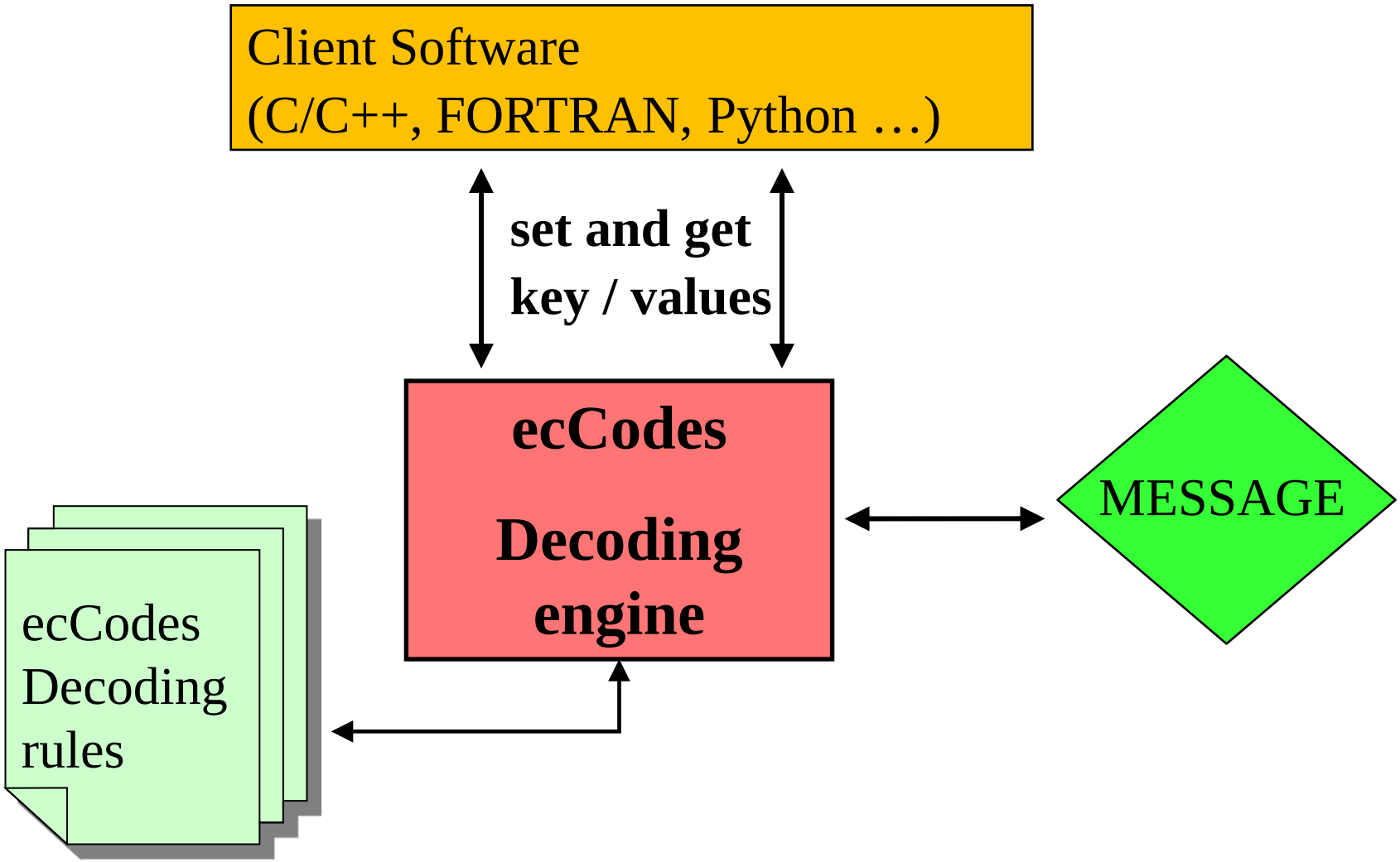
ECMWF Data acquisition and pre-processing

Chair of WMO Task Team on Aviation XML

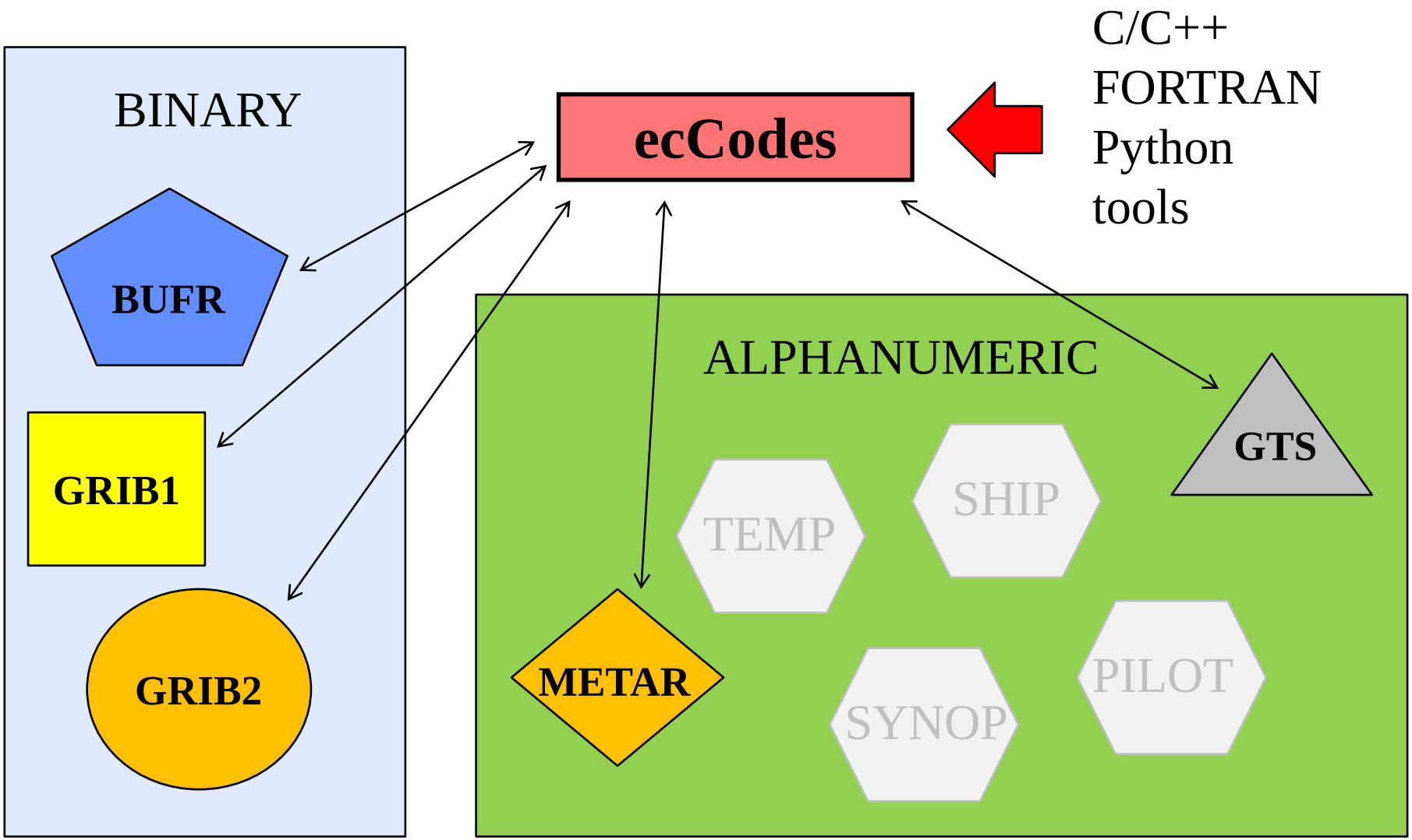
Acquisition and pre-processing at ECMWF



ecCodes


Client Software
(C/C++, FORTRAN, Python …)
set and get
key / values
ecCodes
Decoding
engine
MESSAGE
ecCodes
Decoding
rules

ecCodes


BINARY
BUFR
GRIB1
GRIB2
ecCodes
C/C++
FORTRAN
Python
tools
ALPHANUMERIC
GTS
TEMP
SHIP
METAR
SYNOP
PILOT

METAR

FM 15–XIV METAR **Aerodrome routine meteorological report (with or without trend forecast)**

FM 16–XIV SPECI **Aerodrome special meteorological report (with or without trend forecast)**

CODE FORM:

{METAR or SPECI} COR CCCC YYGGgg**Z** NIL AUTO ddd**ff**G**f**$_m$**f**$_m$ {KMH or KT or MPS} $d_n d_n d_n V d_x d_x d_x$

{VVVV or VVVVNDV or CAVOK} $V_N V_N V_N V_N D_v$ {$RD_RD_R/V_RV_RV_RV_R$i or $RD_RD_R/V_RV_RV_RV_RVV_RV_RV_RV_R$i} w´w´ {$N_sN_sN_sh_sh_sh_s$ or **VV**$h_sh_sh_s$ or NSC or NCD}

T´T´/T´$_d$T´$_d$ **QP**$_H$**P**$_H$**P**$_H$**P**$_H$ **RE**w´w´ {WS RD_RD_R or WS ALL RWY} (WT_sT_s/SS´) ($RD_RD_R/E_RC_Re_Re_RB_RB_R$)

{(TTTTT or NOSIG)} TTGGgg ddd**ff**G**f**$_m$**f**$_m$ {KMH or KT or MPS} {VVVV or CAVOK} {w´w´ or NSW} {$N_sN_sN_sh_sh_sh_s$ or **VV**$h_sh_sh_s$ or NSC}

(RMK .....)

Notes:

(1) METAR is the name of the code for an aerodrome routine meteorological report. SPECI is the name of the code for an aerodrome special meteorological report. A METAR report and a SPECI report may have a trend forecast appended.

(2) The groups contain a non-uniform number of characters. When an element or phenomenon does not occur, the

World Meteorological Organization
Weather • Climate • Water
WMO-No. 306

METAR

- **METAR LIRF 022350Z 33004KT CAVOK 07/05 Q1006 TEMPO 35018G28KT 4000 SHRA BKN014=**
- **METAR EDDM 022350Z 27015KT 9999 -SN FEW012 BKN018 M00/M02 Q1014 TEMPO BKN012=**

ecCodes message definition language

```
if (substr(group,0,1) is "Q" ) {
        qnh=to_string(g,1,4);
        qnhUnits=“hPa”;
}
```

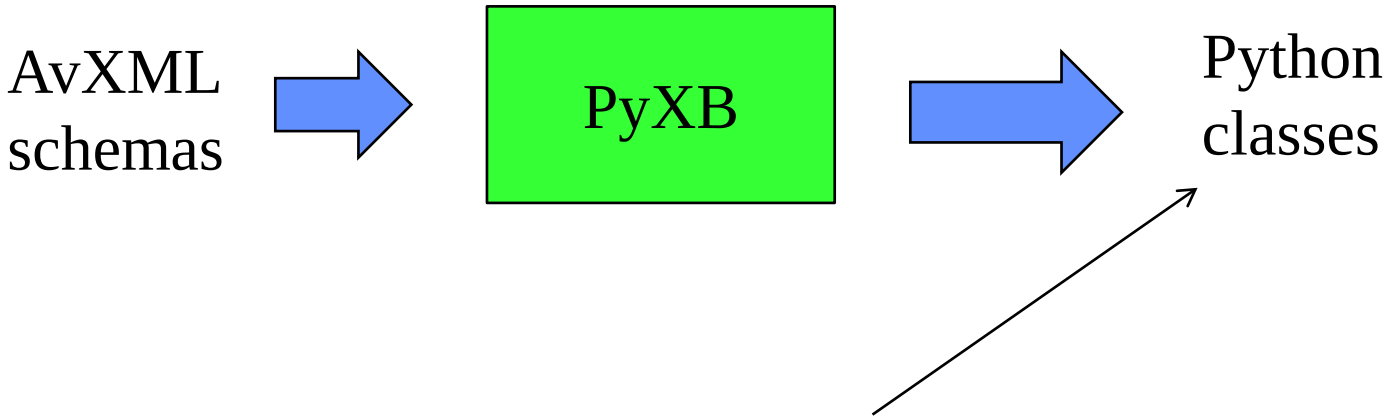
ecCodes python

```
qnh=metar.get("qnh")

temperature=metar.get("temperature")
dewPointTemperature=metar.get("dewPointTemperature")

windSpeed=metar.get("windSpeed")
windDirection=metar.get("windDirection")
windUnits=metar.get("windUnits")
```

PyXB (Python Xml Bindings)



support bi-directional
conversion
between XML documents and
Python objects

PyXB AvXML

```
temperature=metar.get("temperature")
observationRecord.airTemperature=
    _metbasic.AirTemperatureType(temperature)

dewPointTemperature=metar.get("dewPointTemperature")
observationRecord.dewpointTemperature=
    _metbasic.DewPointTemperatureType(dewPointTemperature)
```

PyXB AvXML

windSpeed=metar.get("windSpeed")

observationRecord.surfaceWind.AerodromeSurfaceWind.windSpeed = _metbasic.WindSpeedType(windSpeed)

windDirection=metar.get("windDirection")

observationRecord.surfaceWind.AerodromeSurfaceWind.meanWindDirection=_metbasic.WindDirectionType(windDirection)

windUnits=metar.get("windUnits")

observationRecord.surfaceWind.AerodromeSurfaceWind.windSpeed.uom=windUnits

PyXB AvXML

layer=metarSpeci.CloudLayerPropertyType(metarSpeci.CloudLayerType())

layer.CloudLayer.amount=_metbasic.CloudAmountReportedAtAerodromeType()

layer.CloudLayer.amount.href="http://data.wmo.int/def/bufr-0-20-008/%d"%cloudsCode[i]
layer.CloudLayer.amount.title=cloudsTitle[id]

layer.CloudLayer.base=_metbasic.CloudBaseHeightType(cloudsBase[i])

layer.CloudLayer.base.uom="ft“

observationRecord.cloud.AerodromeObservedClouds.append(layer)

ECMWF

PyXB AvXML

metarXML.observation.OM_Observation.result
.append(observationRecord)

XML serialisation and schema validation

metarXML.toXML()

Results

- **Decoding of ~170 000 METAR with ecCodes takes ~ 2 minutes on a PC. Caching of the decoding rules is very effective on the performance.**
- **Decoding, converting and validating of METAR to AvXML (IWXXM) takes ~ 20 minutes on the same machine. PyXB seams not to be very efficient, or the complexity of the schema is affecting performance.**
- **92% of messages are successfully decoded. 8% unable to locate the airport.**
- **50% of messages are successfully converted and validated. Coding of missing values is not allowed in IWXXM RC1.**
- **Size of the converted messages is ... times the original METAR**

Conclusions

- **METAR decoder can only be written interpreting the regulations. Manual error prone process, long implementation time. Support from ecCodes rules language makes the process quicker.**
- **AvXML decoder/encoder is automatically built from the XML schemas. Python classes are obtained. Also for Java and C++ the same process is possible. The same is possible for BUFR and GRIB.**
- **AvXML access to the information requires a good knowledge of the model.**
- **Combination of ecCodes and PyXB provides a quick way of implementing METAR to AvXML converter.**
- **A big portion of the messages received every day cannot be converted because of missing information.**

Questions?