IWXXM Converter using ecCodes and PyXB : An example of use of ecCodes

20/11/2013

Daniel Dieguez Arias

Meteo-France

14th Workshop on meteorological operational systems

METEO FRANCE

Toujours un temps d'avance

Introduction

- Weather information for aviation is moving to XML format

- International Civil Aviation Organization (ICAO) Annex 3
  - Amendment 76: Nov 2013 “*States in a position to do so* should exchange METAR, SIGMET and TAF in a digital form (XML)”
  - Amendment 77: Nov 2016 “METAR, SIGMET and TAF should be exchanged in a digital form”
- Task team Aviation XML (TT-AvXML) has developed a logical data model named IWXXM (ICAO Meteorological Information Exchange Model)
  - Defines XML format for the reports required by ICAO (the equivalent of existing METAR, SIGMET and TAF)

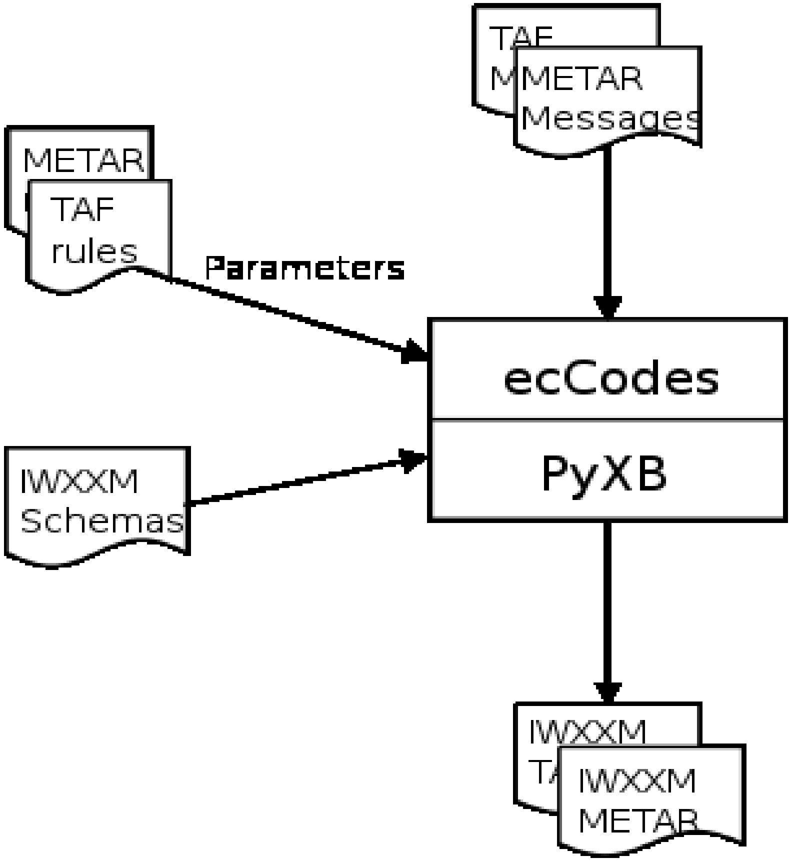
Objective

- Objective :
  - Translate METAR, TAF and SIGMET messages (ascii format) into IWXXM format
- Meteo-France is involved in SESAR activity and format definition due to its operational activity
- ECMWF is involved in format definition and has initiated a software tool named ecCodes
- Therefore Meteo-France and ECMWF have started a collaboration to **develop a tool to convert** METAR, TAF and SIGMET into IWXXM

Converter architecture overview

ECMWF

METEO FRANCE



TAF
M
METAR
Messages

METAR
TAF
rules

Parameters

IWXXM
Schemas

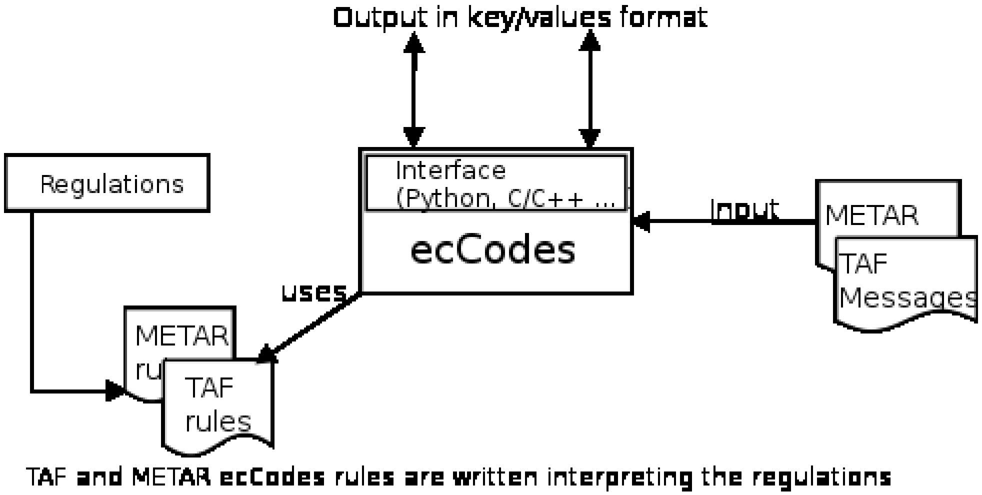
ecCodes

PyXB

IWXXM
T
IWXXM
METAR

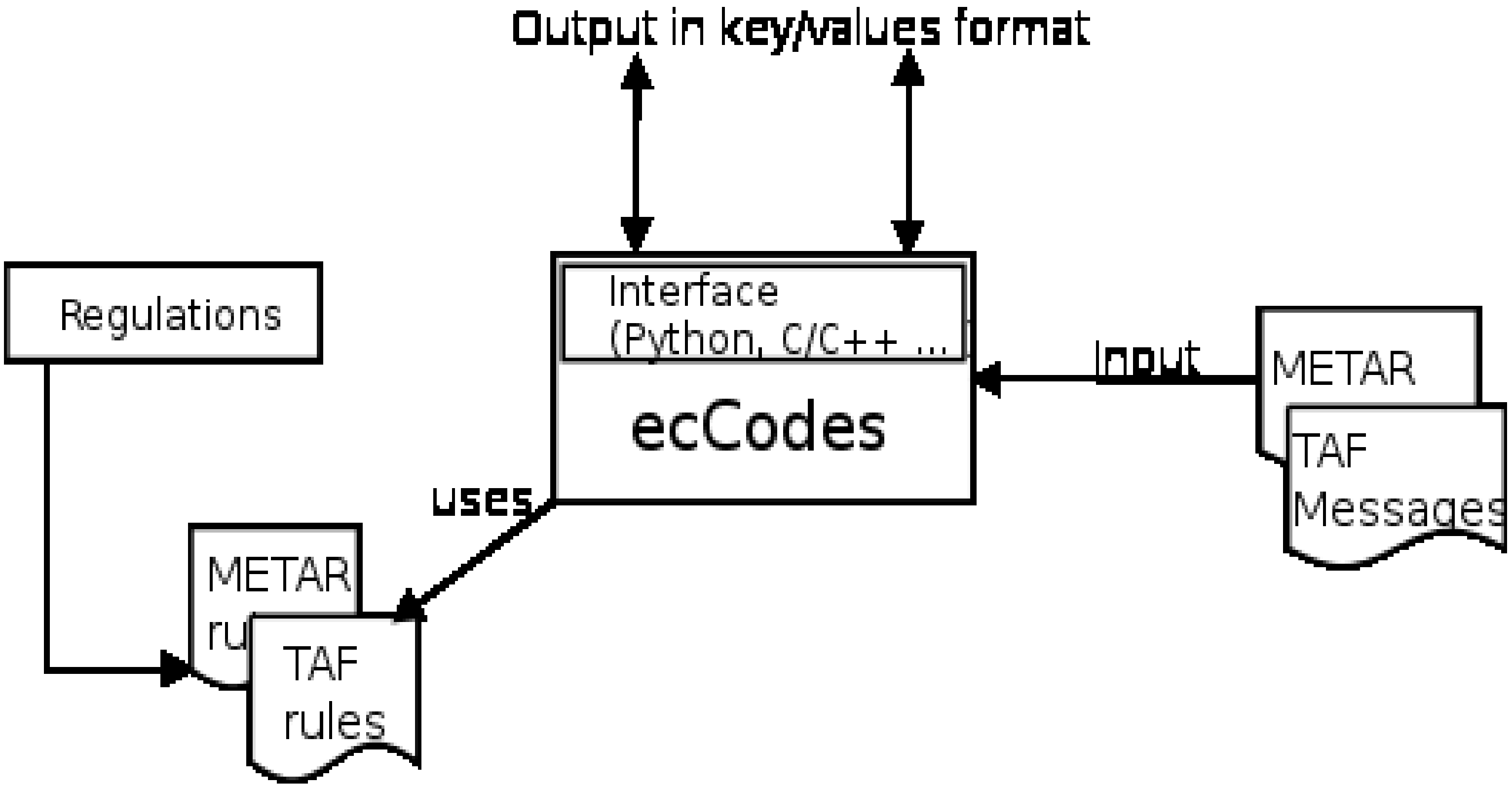
First component

- ecCodes:
  - takes as input TAF and METAR messages
  - uses ecCodes TAF and ecCodes METAR rules to interpret the messages
  - generates key/values representation of the messages applying the rules over the messages
- ecCodes rules are written following the regulations (Annex 3)



TAF and METAR ecCodes rules are written interpreting the regulations

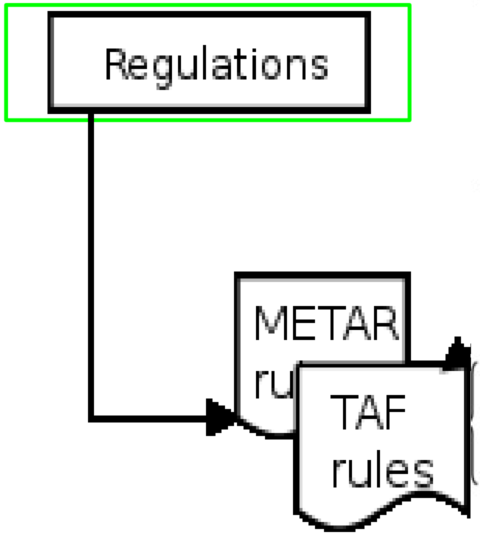
First component

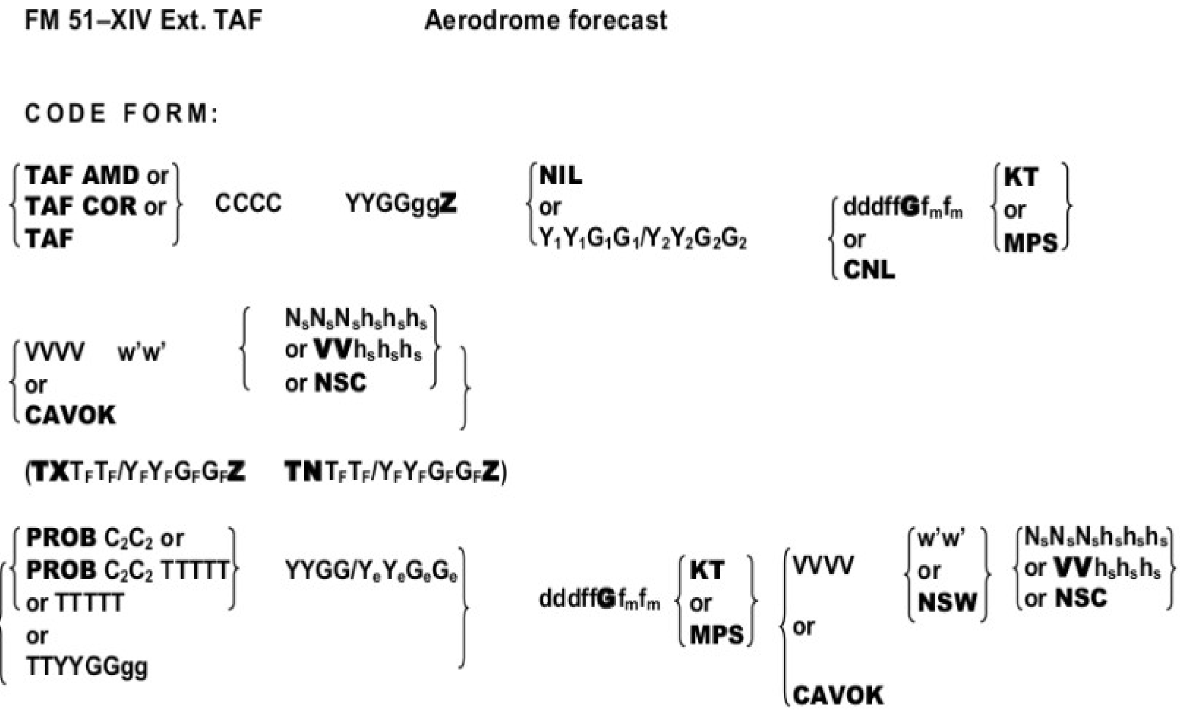


TAF and METAR ecCodes rules are written interpreting the regulations

TAF Regulations

Output in key/values format





FM 51–XIV Ext. TAF Aerodrome forecast

CODE FORM:

{**TAF AMD** or / **TAF COR** or / **TAF**} CCCC YYGGgg**Z** {**NIL** / or / $Y_1Y_1G_1G_1/Y_2Y_2G_2G_2$} {ddd ff**G**$f_mf_m$ / or / **CNL**} {**KT** / or / **MPS**}

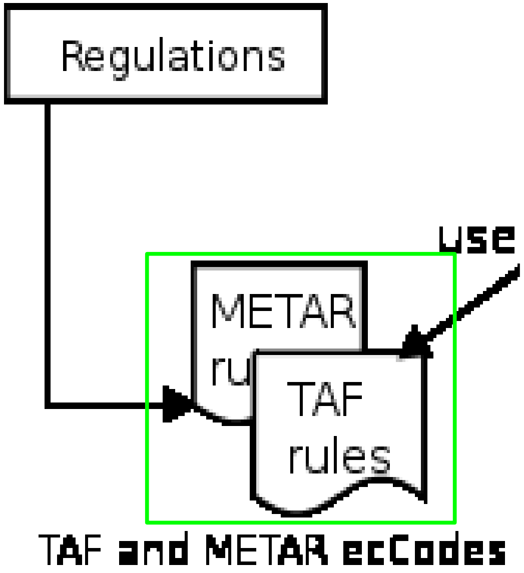
{VVVV w'w' / or / **CAVOK**} {$N_sN_sN_sh_sh_sh_s$ / or **VV**$h_sh_sh_s$ / or **NSC**}

(**TX**$T_FT_F/Y_FY_FG_FG_F$**Z** **TN**$T_FT_F/Y_FY_FG_FG_F$**Z**)

{**PROB** C_2C_2 or / **PROB** C_2C_2 TTTTT / or TTTTT / or / TTYYGGgg} $YYGG/Y_eY_eG_eG_e$ ddd ff**G**f_mf_m {**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**}

TAF and METAR ecCodes rules are written interpreting the regulations

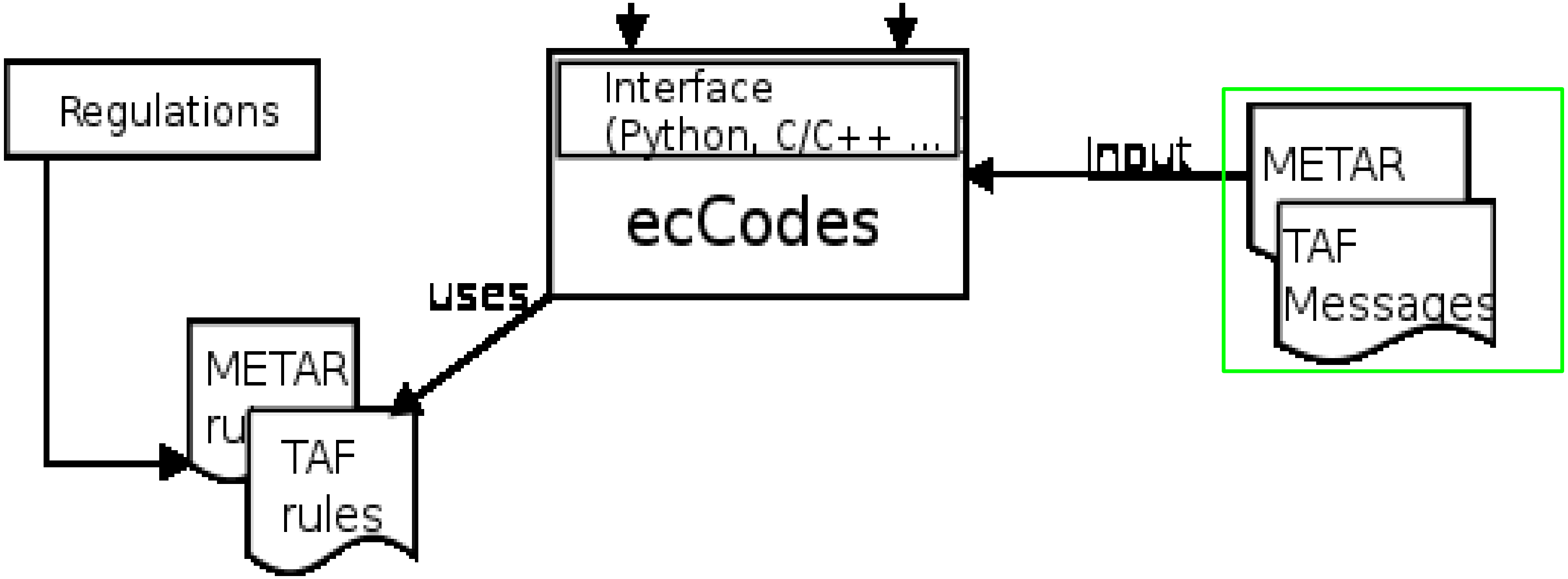
ecCodes TAF rules



```
if (substr(g,-2,2) is "KT") {
  alias windPresent=one;
  constant windUnits="knots" :dump;
  rename(g,windInKnots) ;
  modify windInKnots : hidden;
  if (is_integer(windInKnots,3,2)){
    windSpeed=to_string(windInKnots,3,2) :
     dump;
  }
  if (substr(windInKnots,0,3) is "VRB") {
    constant windDirection= "Variable" : dump;
    alias windVariableDirection = true;
  }
```

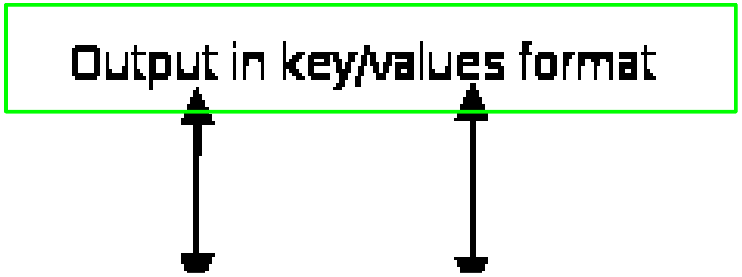
TAF message

TAF EGPD 300458Z 3006/3106 **25008KT** 9999 FEW045 BECMG 3010/3013 19012KT TEMPO 3013/3019 20015G25KT 6000 RA BKN010



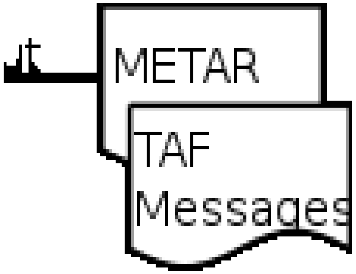
TAF and METAR ecCodes rules are written interpreting the regulations

ecCodes output



Output in key/values format

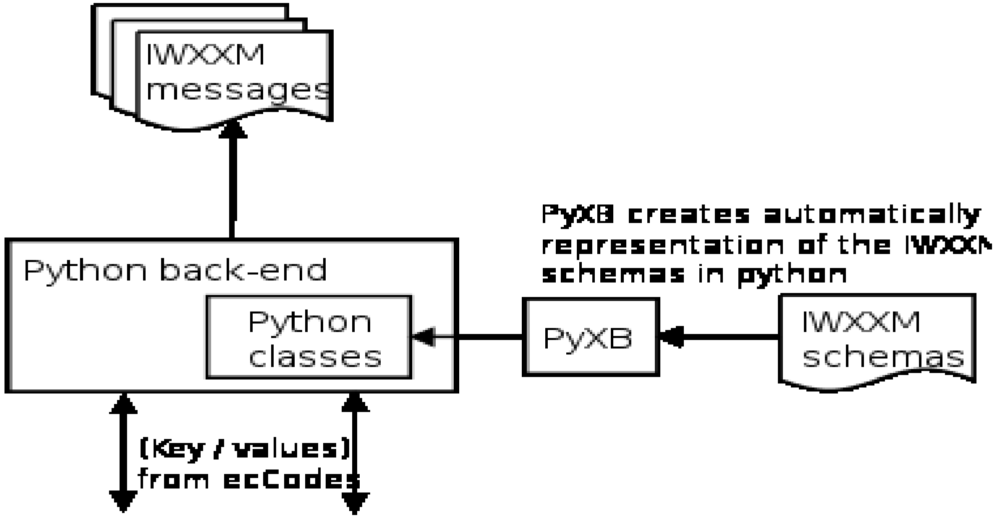
```
#-READ ONLY- dayEndForecastPeriod = 31;
#-READ ONLY- hourEndForecastPeriod = 06;
#-READ ONLY- windUnits = knots;
#-READ ONLY- windSpeed = 08;
#-READ ONLY- windDirection = 250;
#-READ ONLY- clouds1 = FEW045;
#-READ ONLY- prevailingVisibility = 9999;
```



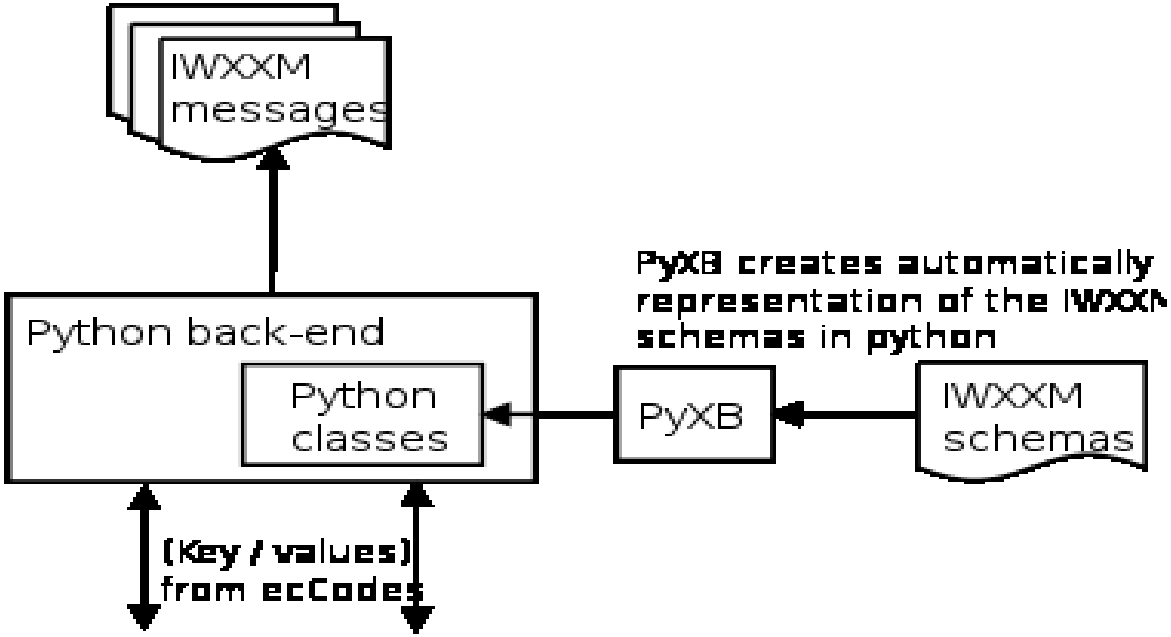
TAF and METAR ecCodes rules are written interpreting the regulations

Second component

- Python back-end:
  - takes as input the key/values from ecCodes
  - uses PyXB (Python XML Schema Binding) to have a python representation of the IWXXM model
    - PyXB provides automatically the IWXXM model Python classes
  - encodes TAF and METAR messages in IWXXM format


IWXXM messages
PyXB creates automatically representation of the IWXXM schemas in python
Python back-end
Python classes
PyXB
IWXXM schemas
(Key / values) from ecCodes

Second component



IWXXM messages

Python back-end

Python classes

PyXB

IWXXM schemas

PyXB creates automatically representation of the IWXXM schemas in python

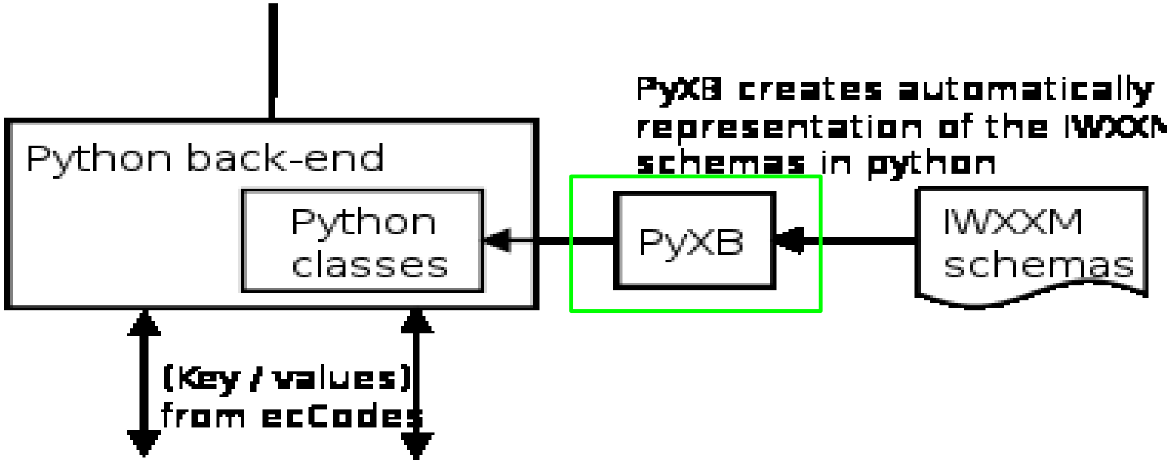
(Key / values) from ecCodes

PyXB

```
surfaceWind=iwxxm.AerodromeSurfaceWindForecastPropertyType(iwxxm.AerodromeSurfaceWindForecastType())

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed=_nsgroup.SpeedType(taf.get_value("windSpeed"))

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed.uom=taf.get_value("windUnits")
```



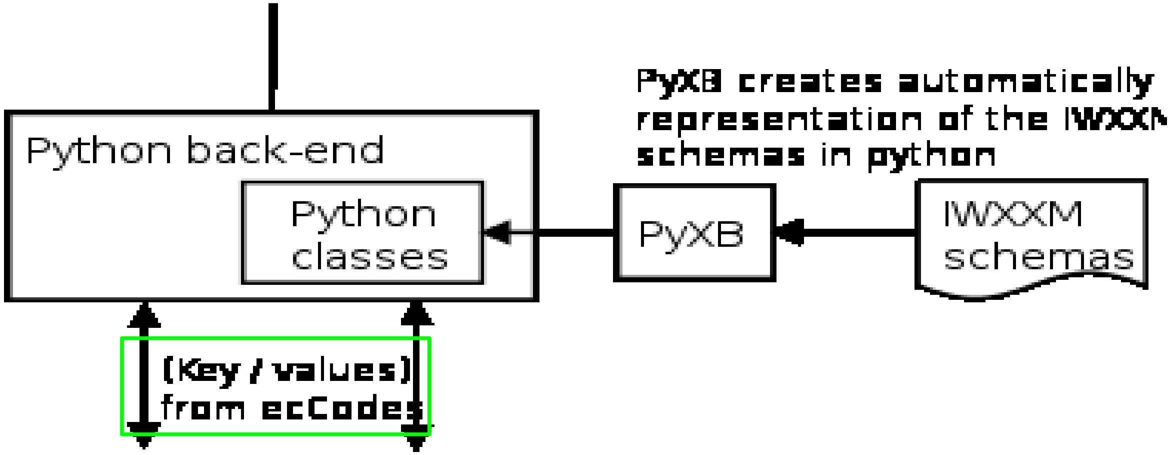
PyXB creates automatically representation of the IWXXM schemas in python

ecCodes key/values output

```
surfaceWind=iwxxm.AerodromeSurfaceWindForecastPropertyType(iwxxm.AerodromeSurfaceWindForecastType())

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed=_nsgroup.SpeedType(taf.get_value("windSpeed"))

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed.uom=taf.get_value("windUnits")
```

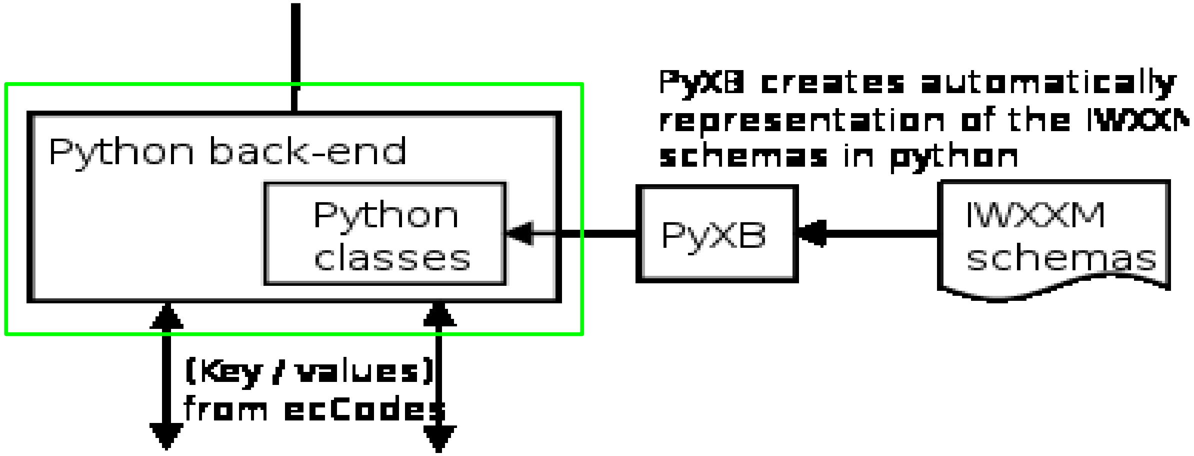

PyXB creates automatically representation of the IWXXM schemas in python
Python back-end
Python classes
PyXB
IWXXM schemas
(Key / values) from ecCodes

Python back-end

```
surfaceWind=iwxxm.AerodromeSurfaceWindForecastPropertyType(iwxxm.AerodromeSurfaceWindForecastType())

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed=_nsgroup.SpeedType(taf.get_value("windSpeed"))

surfaceWind.AerodromeSurfaceWindForecast.meanWindSpeed.uom=taf.get_value("windUnits")
```



PyXB creates automatically representation of the IWXXM schemas in python

Python back-end

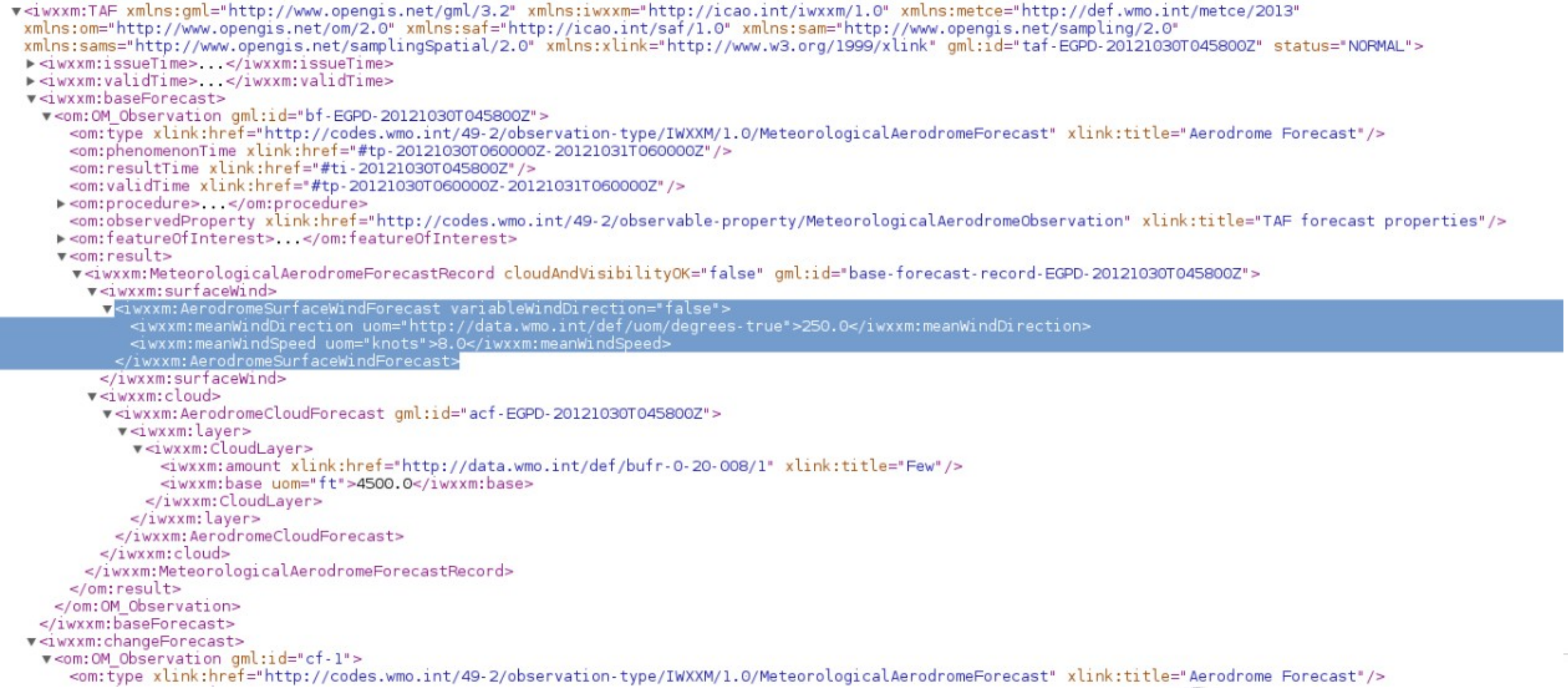
Python classes

PyXB

IWXXM schemas

(Key / values) from ecCodes

IWXXM message



```
▼<iwxxm:TAF xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:iwxxm="http://icao.int/iwxxm/1.0" xmlns:metce="http://def.wmo.int/metce/2013"
 xmlns:om="http://www.opengis.net/om/2.0" xmlns:saf="http://icao.int/saf/1.0" xmlns:sam="http://www.opengis.net/sampling/2.0"
 xmlns:sams="http://www.opengis.net/samplingSpatial/2.0" xmlns:xlink="http://www.w3.org/1999/xlink" gml:id="taf-EGPD-20121030T045800Z" status="NORMAL">
 ▶<iwxxm:issueTime>...</iwxxm:issueTime>
 ▶<iwxxm:validTime>...</iwxxm:validTime>
 ▼<iwxxm:baseForecast>
   ▼<om:OM_Observation gml:id="bf-EGPD-20121030T045800Z">
      <om:type xlink:href="http://codes.wmo.int/49-2/observation-type/IWXXM/1.0/MeteorologicalAerodromeForecast" xlink:title="Aerodrome Forecast"/>
      <om:phenomenonTime xlink:href="#tp-20121030T060000Z-20121031T060000Z"/>
      <om:resultTime xlink:href="#ti-20121030T045800Z"/>
      <om:validTime xlink:href="#tp-20121030T060000Z-20121031T060000Z"/>
     ▶<om:procedure>...</om:procedure>
      <om:observedProperty xlink:href="http://codes.wmo.int/49-2/observable-property/MeteorologicalAerodromeObservation" xlink:title="TAF forecast properties"/>
     ▶<om:featureOfInterest>...</om:featureOfInterest>
     ▼<om:result>
       ▼<iwxxm:MeteorologicalAerodromeForecastRecord cloudAndVisibilityOK="false" gml:id="base-forecast-record-EGPD-20121030T045800Z">
         ▼<iwxxm:surfaceWind>
           ▼<iwxxm:AerodromeSurfaceWindForecast variableWindDirection="false">
              <iwxxm:meanWindDirection uom="http://data.wmo.int/def/uom/degrees-true">250.0</iwxxm:meanWindDirection>
              <iwxxm:meanWindSpeed uom="knots">8.0</iwxxm:meanWindSpeed>
            </iwxxm:AerodromeSurfaceWindForecast>
          </iwxxm:surfaceWind>
         ▼<iwxxm:cloud>
           ▼<iwxxm:AerodromeCloudForecast gml:id="acf-EGPD-20121030T045800Z">
             ▼<iwxxm:layer>
               ▼<iwxxm:CloudLayer>
                  <iwxxm:amount xlink:href="http://data.wmo.int/def/bufr-0-20-008/1" xlink:title="Few"/>
                  <iwxxm:base uom="ft">4500.0</iwxxm:base>
                </iwxxm:CloudLayer>
              </iwxxm:layer>
            </iwxxm:AerodromeCloudForecast>
          </iwxxm:cloud>
        </iwxxm:MeteorologicalAerodromeForecastRecord>
      </om:result>
    </om:OM_Observation>
  </iwxxm:baseForecast>
 ▼<iwxxm:changeForecast>
   ▼<om:OM_Observation gml:id="cf-1">
      <om:type xlink:href="http://codes.wmo.int/49-2/observation-type/IWXXM/1.0/MeteorologicalAerodromeForecast" xlink:title="Aerodrome Forecast"/>
```

Results

	Number of messages	Converted	Not converted	Runtime
METAR	4000	3349	651	2,23 min
TAF	4000	3506	494	3 min

- 84% (METAR) and 87% (TAF) of messages are successfully converted.
- Not converted messages are classified:
  - Not well-formed messages:
    - Some messages do not respect the regulations
  - Not decoded messages:
    - Bugs in the converter
    - ecCodes rules not yet enough well defined

METEO FRANCE
Toujours un temps d'avance

Conclusions

- Combination of eCodes and PyXB provides a quick way to implement a converter from TAF, METAR to IWXXM
- METAR and TAF decoder can only be written interpreting regulations. Long implementation time is needed although ecCodes rules language makes the process quicker
- Use of IWXXM model in python classes requires a good knowledge of the model
- More work is still necessary to produce a full operational converter
- The tool will be used in SESAR project
- We are open for collaboration

Any Questions?

Please, provide your feedback and comments to

daniel.dieguez-arias@meteo.fr

Thank you