

# **GRIB API: Advanced Topics Part II**

**Shahram Najm  
Development Section  
Forecast Department**

# Overview

- **Parameter database**
- **GRIB1 to GRIB2 conversion**
- **Local configuration**

# GRIB 1

“10 metre U component of wind”

**indicatorOfParameter = 33 [u-component of wind (m/s)]**

**table2Version = 3**

**indicatorOfTypeOfLevel = 105 [Specified height level above ground (m)]**

**level = 10**

# GRIB 2

“10 metre U component of wind”

**discipline = 0** [Meteorological products]

**parameterCategory = 2** [Momentum]

**parameterNumber = 0** [u component of wind (m s-1) ]

**typeOfFirstFixedSurface = 103** [Specified height level above ground (m)]

**scaleFactorOfFirstFixedSurface = 0**

**scaledValueOfFirstFixedSurface = 10**

**typeOfSecondFixedSurface = 255** [Missing ]

**scaleFactorOfSecondFixedSurface = MISSING**

**scaledValueOfSecondFixedSurface = MISSING**

# GRIB 1 ECMWF local coding

“10 metre U component of wind”

**indicatorOfParameter = 165** [10 metre u-component of wind (m/s)]  
**table2Version = 128**

# GRIB API solution

- Use a **VIRTUAL** (computed) key to decouple user level from coding level
- User code is setting/getting a **virtual key** and GRIB API is setting the appropriate coded keys
- Local configuration is available to deal with local codes

# GRIB API parameter

- GRIB API provides some edition independent keys to identify a parameter :
  - **paramId**
  - **shortName**
  - **name**
  - **units**
  - **centre**

# GRIB API parameters (GRIB 1)

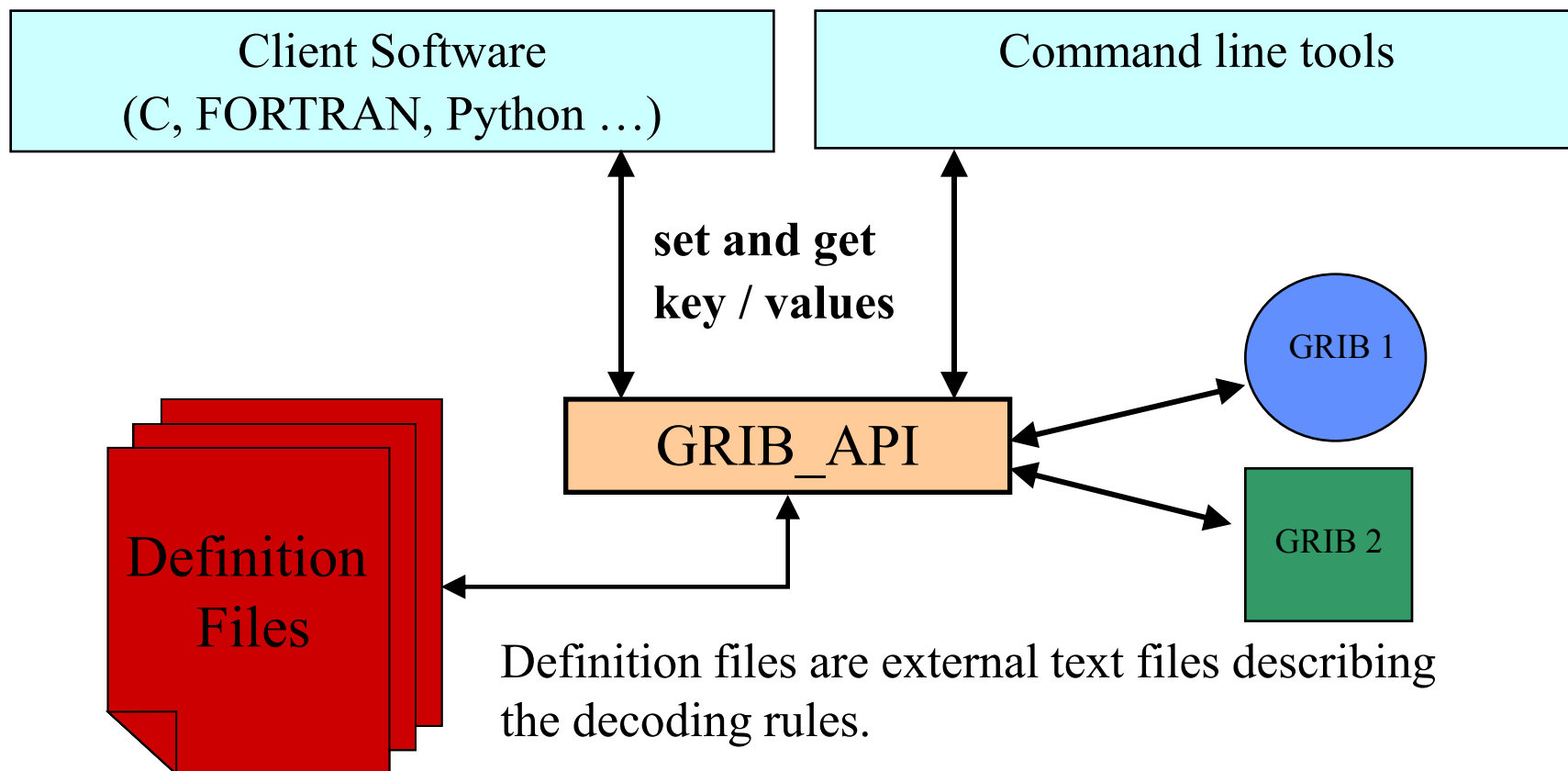
- **centre**
- **table2Version**
- **indicatorOfParameter**
- **levelType**
- **level**
- **...**



# GRIB API parameters (GRIB 2)

- discipline
- parameterCategory
- parameterNumber
- typeOfFirstFixedSurface
- scaleFactorOfFirstFixedSurface
- scaledValueOfFirstFixedSurface
- typeOfSecondFixedSurface
- scaleFactorOfSecondFixedSurface
- scaledValueOfSecondFixedSurface
- productDefinitionTemplateNumber
- ...

# GRIB API - Design



Definition files are external text files describing the decoding rules.

New decoding rules can be added without rebuilding the grib\_api

# Concept

# shortName

## GRIB 2

```
'2t' = { discipline = 0 ; productDefinitionTemplateName = 8 ;  
        parameterCategory = 0 ; parameterNumber = 0 ; typeOfFirstFixedSurface =  
        103 ; scaleFactorOfFirstFixedSurface = 0 ; scaledValueOfFirstFixedSurface =  
        2 ; typeOfSecondFixedSurface = 255 ; scaleFactorOfSecondFixedSurface =  
        missing() ; scaledValueOfSecondFixedSurface = missing(); }
```

## GRIB1

```
'2t' = { indicatorOfParameter=11; table2Version=3;  
        levelType=103; level=2;}
```

## GRIB1 ECMWF local coding

```
'2t' = { indicatorOfParameter=167; table2Version=128;}
```

# Concept

# paramId

## GRIB 2

```
167 = { discipline = 0 ; productDefinitionTemplateName = 8 ;  
        parameterCategory = 0 ; parameterNumber = 0 ; typeOfFirstFixedSurface =  
        103 ; scaleFactorOfFirstFixedSurface = 0 ; scaledValueOfFirstFixedSurface =  
        2 ; typeOfSecondFixedSurface = 255 ; scaleFactorOfSecondFixedSurface =  
        missing() ; scaledValueOfSecondFixedSurface = missing(); }
```

## GRIB1

```
167 = { indicatorOfParameter=11; table2Version=3;  
        levelType=103; level=2;}
```

## GRIB1 ECMWF local coding

```
167 = { indicatorOfParameter=167; table2Version=128;}
```

# Parameters in GRIB API

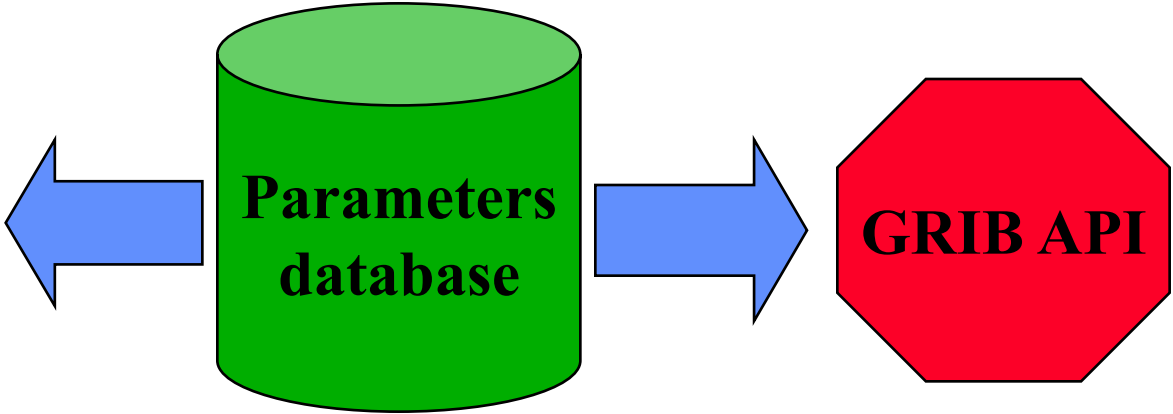
<http://old.ecmwf.int/publications/manuals/d/gribapi/param>

# Parameters database

WEB

ECMWF GRIB API Parameters

parameter ID	name	units
1	Surface temperature	m2t1
2	Sea level pressure	m2t1
3	Relative humidity	%
4	Equivalent potential temperature	K
5	Equivalent potential temperature	K
6	Sea level pressure	hPa
7	Sea level pressure	hPa
8	Sea level pressure	hPa
9	Sea level pressure	hPa
10	Sea level pressure	hPa
11	Sea level pressure	hPa
12	Sea level pressure	hPa
13	Sea level pressure	hPa
14	Sea level pressure	hPa
15	Sea level pressure	hPa
16	Sea level pressure	hPa
17	Sea level pressure	hPa
18	Sea level pressure	hPa
19	Sea level pressure	hPa
20	Sea level pressure	hPa
21	Sea level pressure	hPa
22	Sea level pressure	hPa
23	Sea level pressure	hPa
24	Sea level pressure	hPa
25	Sea level pressure	hPa
26	Sea level pressure	hPa
27	Sea level pressure	hPa
28	Sea level pressure	hPa
29	Sea level pressure	hPa
30	Sea level pressure	hPa
31	Sea level pressure	hPa
32	Sea level pressure	hPa
33	Sea level pressure	hPa
34	Sea level pressure	hPa
35	Sea level pressure	hPa
36	Sea level pressure	hPa
37	Sea level pressure	hPa
38	Sea level pressure	hPa
39	Sea level pressure	hPa
40	Sea level pressure	hPa
41	Sea level pressure	hPa
42	Sea level pressure	hPa
43	Sea level pressure	hPa
44	Sea level pressure	hPa
45	Sea level pressure	hPa
46	Sea level pressure	hPa
47	Sea level pressure	hPa
48	Sea level pressure	hPa
49	Sea level pressure	hPa
50	Sea level pressure	hPa
51	Sea level pressure	hPa
52	Sea level pressure	hPa



# Parameters: Practicals

## To get the practicals:

```
tar xvf ~trx/grib_api/grib_parameters.tar
```

1. You have two grib messages start.grib1 and start.grib2.
2. Create the file 10u.grib1 setting shortName=10u in start.grib1
3. Create the file 10u.grib2 setting shortName=10u in start.grib2
4. Do `grib_ls -n parameter 10u.grib1 10u.grib2` . Do you see any difference.
5. Compare the `grib_dump -O` of the two files and of the two messages in each file.
6. Set shortName=10v.

# GRIB1 to GRIB2 conversion

```
grib_set -s edition=2 in.grib1 out.grib2
```

## conversion of

- time
- geography
- vertical
- parameter
- local
- data



# GRIB1 to GRIB2 conversion

- **Parameter conversion is particularly complex due to the difference between the two coding standards and the local tables used by some meteorological centres.**
- **The conversion is based on the parameter's unique identifier paramId.**

# paramId based conversion

- How to produce a GRIB for a “2 metre temperature”

```
grib_set -s paramId=165 in.grib1 out.grib1
```

```
grib_set -s paramId=165 in.grib2 out.grib2
```

- How to convert a GRIB1 to GRIB2

```
grib_set -s edition=2 in.grib1 out.grib2
```

- During the conversion to edition=2 GRIB API is copying the paramId value from the GRIB1 to the GRIB2:

1. get paramId(=165) from GRIB1
2. change edition to 2 producing a GRIB2
3. set paramId(=165) in GRIB2

# paramId based conversion

- The conversion is possible only if a paramId is defined for both editions.
- Check on the [parameters web](#) page if a conversion is possible.

## **grib1to2**

- **For user convenience a tool to convert model level data is provided.**
- **Provides conversion of ECMWF model level data from edition 1 to 2.**
- **The only difference between grib1to2 and “grib\_set -s edition=2” is that grib1to2 works only on model level data and returns an error code on other level types.**

# Parameters: Practicals

We refer to the same files produced in the previous practical

1. Convert 10u.grib1 to its GRIB2 version 10u\_converted.grib2.
2. Do `grib_ls -n parameter 10u.grib2`
3. Do `grib_ls -n parameter 10u_converted.grib2`
4. Take the first message from start.grib1 and save it to ecmf.grib1
5. Set the paramId of ecmf.grib1 to 162089. Save it as ecmf.162089.grib1
6. Convert ecmf.162089.grib1 to GRIB edition 2. Why does it fail?

# Local configuration

- The external text files defining the decoding rules used by the decoding engine are called **definition files**.
- For each installation there is a default set of definition files.
- The `GRIB_DEFINITION_PATH` environment variable can be set to use local definition files instead of the definition files provided within the distribution.

# Local configuration

- The parameter descriptions for a given “centre” are contained in the files **shortName.def**, **paramId.def**, **units.def**, **name.def** in the directories

**BASE\_DIR/definitions/grib1/localConcepts/[centre:s]**

**BASE\_DIR/definitions/grib2/localConcepts/[centre:s]**

Note: ‘centre:s’ means the centre as a *string* e.g. ecmf, kwbc, cnmc etc

- The general parameter descriptions are contained in the files **shortName.def**, **paramId.def**, **units.def**, **name.def** in the directories

**BASE\_DIR/definitions/grib1**

**BASE\_DIR/definitions/grib2**

# Local configuration

**GRIB\_DEFINITION\_PATH=/joe/definitions:/grib\_api/definitions**

- **The library searches for each required definition file first in /joe/definitions and then in /grib\_api/definitions.**
- **If the file is found in /joe/definitions then it used by the decoding engine.**
- **The user can override all the definition files with his/her own definition files.**
- **We suggest you only override the definition files containing the parameter information**



# Local configuration: defining a parameter locally

- Get the directory of the definition files with the utility `grib_info`
  - set the environment variable `GRIB_DEFINITION_PATH=local_directory:default_definition_directory`
  - Put in the directories:
    - `local_directory/grib1/localConcepts/[centre:s]`
    - `local_directory/grib2/localConcepts/[centre:s]`
- The files `shortName.def`, `paramId.def`, `name.def` & `units.def`.

# Local configuration: defining a parameter locally

- Example from paramId.def (for GRIB1)

```
#Direction of wind waves
```

```
'500072' = {  
    table2Version = 112;  
    indicatorOfParameter = 101;  
}
```

- Example from shortName.def (for GRIB1)

```
#Total precipitation of at least 10 mm
```

```
'tpg10' = {  
    table2Version = 131;  
    indicatorOfParameter = 62;  
}
```

# Local configuration: Practicals

## To get the practicals:

```
tar xvf ~trx/grib_api/localConfig.tar
```

1. What parameter is contained within the two files x.grib1 and x.grib2?
2. Run grib\_info to find the location of the default definitions
3. Now set GRIB\_DEFINITION\_PATH to include the “mydefs” directory e.g.  

```
export GRIB_DEFINITION_PATH=`pwd`/mydefs:default_defs
```
4. Now see if grib api recognizes the name, units etc
5. Test the GRIB1 to GRIB2 conversion. Compare the output with the provided x.grib2 file
6. Study the directory structure of “mydefs” and its local concept files

**Questions ?**