Integrating meta information in the CAPRI production chain

- Wolfgang Britz and Alexander Gocht -, April 2008 –

Motivation and background

CAPRI is not only a large-scale complex modelling system, but also the work steps from raw data extraction over a calibrated model ex post to a ex-ante baseline are manifold, and the outcome of each step may impact on how the model reacts in a policy experiment. As many of the programs responsible for specific work step may also only started for a sub-set of the data (e.g. only for one or several Member States, only for NUTS 0), it may easily to use data sets in the different work step which reflect different versions of the data – either regarding updates of the underlying statistical raw data, the version of the algorithms employed or the steering options of the programs. Controlling when, who, how and based on what data a certain work step in the production chain was performed would increase transparency and avoid pitfalls as forgetting to run one of the step, and thus using outdated or even erroneous data.

When the Java based GUI was developed, a software solution was integrated which stored meta data about starts of the various GAMS programs in a XML-file, and presented them to the user. The idea was to keep track about actions of the user and to allow to compare the order of the work step and task in the production chain with the dates of execution. Besides struggling with the fact that sometimes the information was updated despite the GAMS process not finishing successfully, results for certainh work steps are now often downloaded from the Software Version system (SVN), and the GUI will in that case not notice that the meta information would need to be updated.

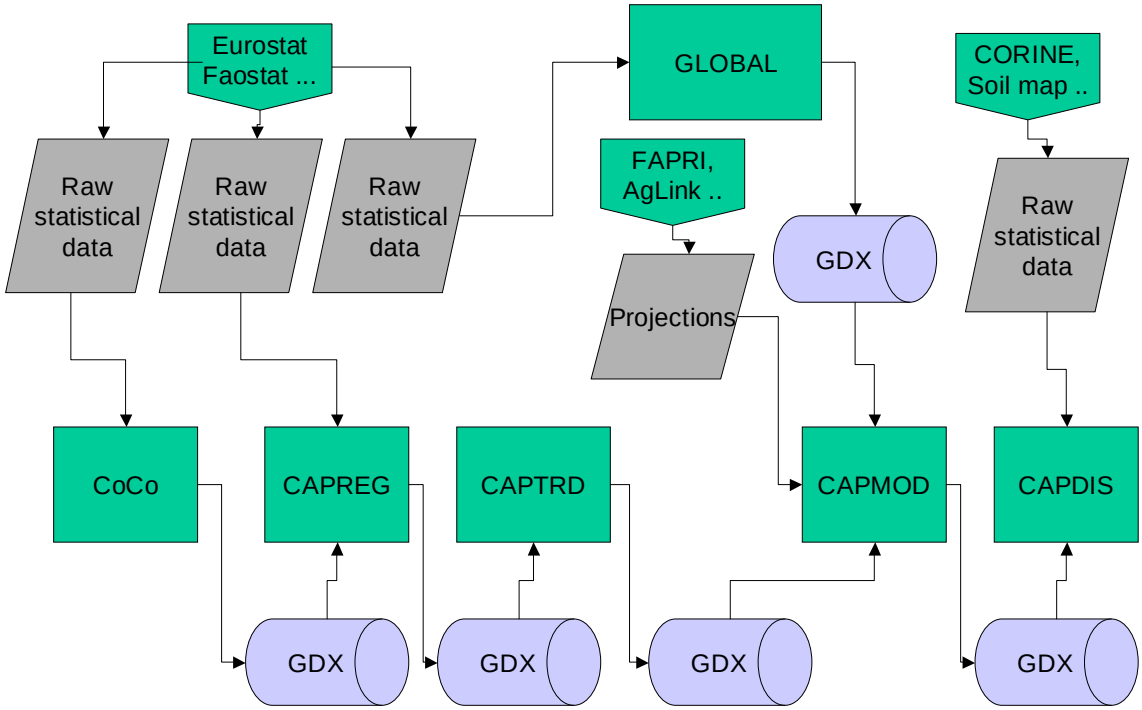
Additionally, once a work step is executed, the information about the previous status is lost albeit existing result sets may still be based on it. That has to change. On top, it seems important to ensure that the meta data information is comprised in the same files as the data themselves. The paper therefore discussed as a successfully implemented solution to handle meta data based on storing the information in the GDX files along with the numerical values.

Table: Old meta data information in the GUI

#	Date	Workstep	Task	Key	Description
0	21.03.2006	Build database	Build national database	COCO	
1	21.03.2006	Build database	Estimate consumer prices	COCO	
7	31.08.2006	Generate baseline	Generate policy shifts	POLSHIFT	
6	03.09.2006	Generate baseline	Generate expost results	EXPOST	
11	04.09.2006	Run simulation	Run simulation	BASELINE	Fully iterated Ba
11	04.09.2006	Run simulation	Run simulation	MTRAGD	AGENDA 2000 p
11	04.09.2006	Run simulation	Run simulation	MTRFDC	Full decoupling
11	04.09.2006	Run simulation	Run simulation	WTOHRB	Harbinson prop
11	05.09.2006	Run simulation	Run simulation	BASELINE	Fully iterated Ba
11	05.09.2006	Run simulation	Run simulation	EUMEDP	Partial liberalisa

A new approach

Graphic: Principal data flow in the CAPRI production chain



The work step flow of CAPRI is based on the I/O of so-called GDX files, an internal binary data format of GAMS as seen below. CoCo (Completeness & Consistency) is a mostly statistically based approach where data from different statistical domains and data providers are integrated into a time series data base at national level which covers mainly market balances, unit value prices, the position of the Economic Accounts for Agriculture, acreage and herd sizes as well as output coefficients. It is the common data base for both CAPSIM and CAPRI. Next in the production chain is CAPREG, responsible of adding the regional breakdown of herd sizes, acreages and crop yields to the time series provided by CoCo, but also input coefficients, crop nutrient requirements and related fertilizer application rates and animal requirement and feeding coefficients. It also derive manifold economic and environmental indicators from the results. Both CoCo or CAPREG may be started for single Member States, or a different set of years, and CAPREG also for different base years. That flexibility has proven extremely useful to avoid being forced to always update the information for 27 EU Member states, Norway and Western Balkan countries even if only some data for a single countries needs a revision. But the flexibility clearly carries the risk to forget to run the program after a data base update or a change in the code for some of the countries. As CoCO and CAPREG produce huge file comprising data for all countries together, the only information so far was provided by the file system on when the file was stored the last time.

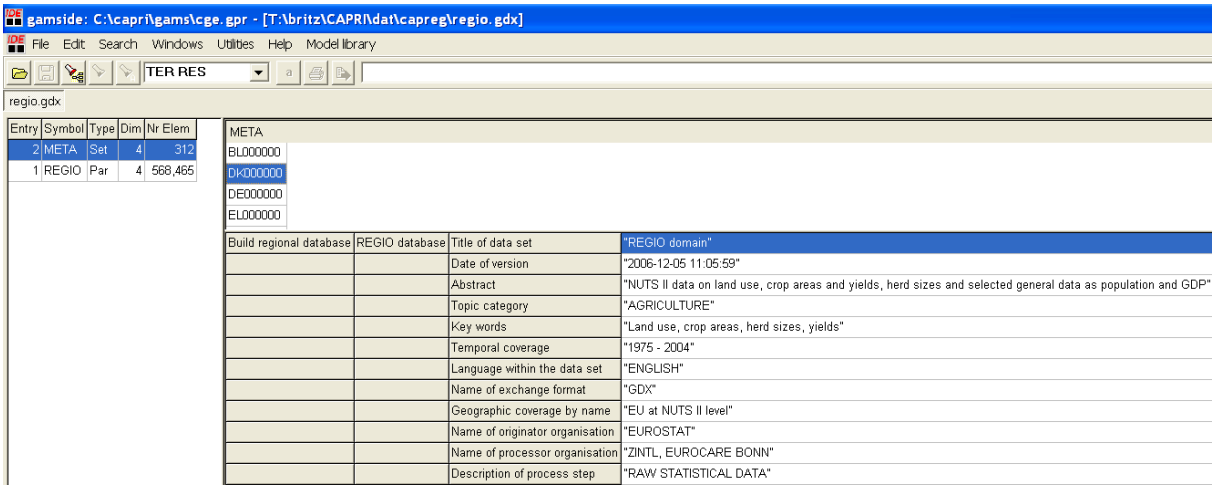
The regional and national time series from CAPREG enter the projection engine CAPTRD, where they together with projections from other studies and experts feed into a set of mutually consistent data on market balances, acreages and herd sizes, yields and other output coefficients and feed coefficients for a future simulation year. As with CoCo and CAPREG, CAPTRD may be started for single countries, and at the NUTS 0 or NUTS II level.

A calibration of the simulation engine CAPMOD requires additional data for other parts of the world and bi-lateral trade flows in quantities and values, data which are processed by GLOBAL.

All these processes store the numerical results in GDX files, and consecutive steps read them from the GDX files as input. It seemed therefore most promising to introduce meta information in the GDX files themselves. The big advantage would be that shipping the data from one workstation to another or to and from the SVN server would also automatically move the meta information along. On top, the meta information could be also processed and viewed with regular GAMS tools as seen below.

The main challenge consists in the fact that meta information requires in some cases strings to pass information e.g. about the user. “Missing” the double precision numerical data stored in parameters in GAMS for meta data is therefore cumbersome. The GDX format allows however to store and load GAMS SETs which are collection of strings – a unique string to identify the set element and an attached long text. SETS in GAMS can be multi-dimensional and are therefore flexible enough to host the necessary meta information.

Meta data as a cross set in GDX file and viewed by the GAMS IDE



In order to generate the meta information in a uniform standard and to ensure that it is passed to the programs, it is generated automatically by the CAPRI GUI before a GAMS process is

called, or in case where data sets so far had been generated by other means, the meta information had been added manually. It is therefore even more important then in the past that users use the GUI to run work steps in the production chain, as otherwise, the meta information will most probably not correctly updated. The example below shows the information passed in the current prototype solution.

Example of meta information stored in a GAMS set

```
SET META  /
DK000000.'Build regional database'.'Build regional database'.'TITLE OF DATA SET' Build regional database CAPREG
DK000000.'Build regional database'.'Build regional database'.WORKSTEP        Build database
DK000000.'Build regional database'.'Build regional database'.KEY             CAPREG
DK000000.'Build regional database'.'Build regional database'.'NAME OF PROCESSOR ORGANISATION' Britz
DK000000.'Build regional database'.'Build regional database'.'TEMPORAL COVERAGE' 1984 - 2004
DK000000.'Build regional database'.'Build regional database'.'DATE OF VERSION' 2008-04-29 15:39:22
DK000000.'Build regional database'.'Build regional database'.BASEYEAR
DK000000.'Build regional database'.'Build regional database'.SIMYEAR
DK000000.'Build regional database'.'Build regional database'.MODEL_SWITCHES
DK000000.'Build regional database'.'Build regional database'.MEMBER_STATES   DK
DK000000.'Build regional database'.'Build regional database'.'LANGUAGE WITHIN THE DATA SET'          ENGLISH
DK000000.'Build regional database'.'Build regional database'.'NAME OF EXCHANGE FORMAT'               GDX
DK000000.'Build regional database'.'Build regional database'.'NAME OF OWNER ORGANISATION'            CAPRI network
DK000000.'Build regional database'.'Build regional database'.'NAME OF ORIGINATOR ORGANISATION'       CAPRI network
DK000000.'Build regional database'.'Build regional database'.'DESCRIPTION OF PROCESS STEP' 'Build database, Build regional database'
DK000000.'Build regional database'.'Build regional database'.'GEOGRAPHIC COVERAGE BY NAME' DK
/
```

The information comprises in a such set can be stored in a GDX file, preserving the long texts shown in the right hand side, and loaded into a GAMS program or, via a library, accessed by higher programming languages.

In the production chain, each program will load along with the numerical data provided by upstream work steps the meta information linked to it, and add its own. In a simulation result set, it is therefore possible to check when and by whom the national time series from CoCo had been generated etc. When information from the SVN server about the version of the underlying code is integrated as well, such an approach will greatly improve the transparency in the CAPRI production chain, and increase the confidence when comparing policy experiments.

Meta data standard implemented

Hazeu et.al. 2006 proposed a specific set of meta data to be commonly implemented in the SEAMLESS and SENSOR integrated projects, and some major data bases both from EUROSTAT as well as CAPREG and COCO outcomes were documented according that proposal. The basic set proposed comprises the following attributes:

Issue Required Iso code
Title * 15.24.360
Metadata on metadata
Point of contact:
Name of contact organisation * 8.376
Name of contact person * 8.375

Position of contact person 8.377
Role of organisation 8.379
Address: Delivery point * 8.378.389.381
Address: City * 8.378.389.382
Address: Province, state * 8.378.389.383
Address: Postal code * 8.378.389.384
Address: Country * 8.378.389.385
Address: E-mail * 8.378.389.386
Weblink *
Last modified * 9
Name of standard 10
Version of standard 11

Data set identification:

Title of the data set * 15.24.360
Alternative title * 15.24.361
Abstract * 15.25
Keywords * 15.33.53
Topic category * 15.41
Temporal coverage *
Version of data set * 15.24.363
Date of version * 15.24.362.394

Reference system:

Name of reference system (*) 13.196.207
Datum name (*) 13.192.207

Ellipsoid:

Name of ellipsoid (*) 13.191.207
Semi-major axis (*) 13.193.202
Axis units (*) 13.193.203
Flattering ratio (*) 13.193.204

Projection:

Name of projection (*) 13.190.207
Standard parallel (*) 13.194.217
Longitude of central meridian (*) 13.194.218
Latitude of projection origin (*) 13.194.219
False easting (*) 13.194.220
False northing (*) 13.194.221
False easting northing units (*) 13.194.222
Scale factor at equator (*) 13.194.223
Longitude of projection centre (*) 13.194.224
Latitude of projection centre (*) 13.194.225

Distribution information:

Owner:

Name of owner organisation * 15.29.376
Name of contact person 15.29.375
Position of contact person 15.29.377
Role of owner organisation 15.29.379
Address: Delivery point 15.29.378.389.381
Address: City 15.29.378.389.382
Address: Province, state 15.29.378.389.383
Address: Postal code 15.29.378.389.384
Address: Country 15.29.378.389.385

Address: E-mail 15.29.378.389.386
Originator:
Name of originator organisation 15.29.376
Name of contact person
Position of contact person
Role of originator organisation
Address: Delivery point
Address: City
Address: Province, state
Address: Postal code
Address: Country
Address: E-mail
Processor:
Name of processor organisation
Name of contact person
Position of contact person
Role of processor organisation
Address: Delivery point
Address: City
Address: Province, state
Address: Postal code
Address: Country
Address: E-mail
Distributor:
Name of distributor organisation
Name of contact person
Position of contact person
Role in distributor organisation
Address: Delivery point
Address: City
Address: Province, state
Address: Postal code
Address: Country
Address: E-mail
On-line delivery
Access rights:
Type of constraint 20.70
Description of restriction 20.72
Other information:
Language within the data set * 15.39
Exchange format:
Name of exchange format * 15.32.285
Version of exchange format * 15.32.286
Methodology description: 18.81.83
Link to methodological report
Changes since last version
Process steps:
Description of process steps 18.81.84.87
Resource name 18.81.84.91.360
Resource date 18.81.84.91.362
Scale * 15.38.60.57

Geographic accuracy 15.38.60.57
Geographic box: 15.38.61
West bound longitude (*) 15.45.336.344
East bound longitude (*) 15.45.336.345
South bound latitude (*) 15.45.336.346
North bound latitude (*) 15.45.336.347
Geographic coverage by name *
List of attributes
Data type (vector / raster)

It is quite clear that for the majority of the data sources handled by CAPRI and the generated data sets, many of the proposed attributes are not relevant, especially those needed for georeferenced data sets (everything under reference system, many items under other information). Equally, implementation of metadata on metadata may at current stage be less important. Therefore, the following standard attributes were selected:

Title of the data set * 15.24.360
Abstract * 15.25
Keywords * 15.33.53
Topic category * 15.41
Temporal coverage *
Date of version * 15.24.362.394
Name of owner organisation * 15.29.376
Name of originator organisation 15.29.376
Name of distributor organisation
Description of process steps 18.81.84.87
Language within the data set * 15.39
Name of exchange format * 15.32.285

Further elements may be added where thought necessary. That standard set is handled explicitly by the programs, in the sense that e.g. meta data referring to data for countries not comprised in a scenario run are deleted.

Technical implementation

The technical implementation relates to three questions:

1. Introduction of meta data in the different data files to document ingoing data of various origins (raw statistical data, estimations etc.)
2. Introduction of meta data for data sets generated by GAMS programs started via the GUI.
3. Handling of meta data inside the GAMS code, and later during exploitation.

Introduction of meta data into data files

In that case, the information must be edited by hand. The following code fragment shows an example for the REGIO data base from EUROSTAT:

```
SET META  /
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'NAME OF EXCHANGE FORMAT'         GDX
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'TITLE OF DATA SET'              'REGIO domain'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'ABSTRACT'                       'NUTS II data on land use, crop areas and yields, herd sizes and selected general data as population and GDP'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'TOPIC CATEGORY'                 'AGRICULTURE'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'KEY WORDS'                      'Land use, crop areas, herd sizes, yields'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'NAME OF ORIGINATOR ORGANISATION' EUROSTAT
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'LANGUAGE WITHIN THE DATA SET'   ENGLISH
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'GEOGRAPHIC COVERAGE BY NAME'     'EU at NUTS II level'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'TEMPORAL COVERAGE'              1975 - 2004
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'DESCRIPTION OF PROCESS STEP'    'RAW STATISTICAL DATA'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'NAME OF PROCESSOR ORGANISATION' 'ZINTL, EUROCARE BONN'
 (SET.MS_EU15).'Build regional database'.'REGIO database'.'DATE OF VERSION'                2006-12-05 11:05:59
/;
```

An important feature in that respect that the possibility to generate longtexts simultaneously for vectors of codes (in the example the SET.MS_EU15), renders the statement rather compact.

However, manal editing is always a both cumbersome and error prone process. It is therefore advisable to rapidly implement conversion routines from raw statistical data into GAMS code in JAVA, to integrate them in the GUI and to introduce the meta data automatically into the data files. However, there will always be a number of files (e.g. data from ESIM, FAPRI, from the international fertilizer association etc.) where meta data need to be added manually. But once all files had been documented once, updating of the meta data is relatively straightforward.

Introduction of meta data for generated data sets

The technical implementation is based in the general approach in CAPRI. The JAVA based GUI will based along with other steering information the meta information about the run in GAMS format to a work step as shown above. The necessary information are collected from user input.

```
SET META  /
DK000000.'Build regional database'.'Build regional database'.'TITLE OF DATA SET' Build regional database CAPREG
DK000000.'Build regional database'.'Build regional database'.WORKSTEP        Build database
DK000000.'Build regional database'.'Build regional database'.KEY             CAPREG
DK000000.'Build regional database'.'Build regional database'.'NAME OF PROCESSOR ORGANISATION' Britz
DK000000.'Build regional database'.'Build regional database'.'TEMPORAL COVERAGE' 1984 - 2004
DK000000.'Build regional database'.'Build regional database'.'DATE OF VERSION' 2008-04-29 15:39:22
DK000000.'Build regional database'.'Build regional database'.BASEYEAR
DK000000.'Build regional database'.'Build regional database'.SIMYEAR
DK000000.'Build regional database'.'Build regional database'.MODEL_SWITCHES
DK000000.'Build regional database'.'Build regional database'.MEMBER_STATES   DK
DK000000.'Build regional database'.'Build regional database'.'LANGUAGE WITHIN THE DATA SET'          ENGLISH
DK000000.'Build regional database'.'Build regional database'.'NAME OF EXCHANGE FORMAT'              GDX
DK000000.'Build regional database'.'Build regional database'.'NAME OF OWNER ORGANISATION'           CAPRI network
DK000000.'Build regional database'.'Build regional database'.'NAME OF ORIGINATOR ORGANISATION'      CAPRI network
DK000000.'Build regional database'.'Build regional database'.'DESCRIPTION OF PROCESS STEP' 'Build database, Build regional database'
DK000000.'Build regional database'.'Build regional database'.'GEOGRAPHIC COVERAGE BY NAME' DK
/
```

Handling of meta data in GAMS

The work steps will first load the existing meta information from previous work step at compile as shown below. That is necessary to allow redefinition based on the "$ONMULTI" settings to stepwise append further information to the META data set. In order to allow that new meta data overwrite existing one from previous runs, the meta data from older runs need to be loaded first before new meta data is introduced.

Code fragment showing how to load meta data from a GDX file at compile time

```
$if not exist "..\results\capreg\res_%BAS%_.gdx" $goto after
$gdxin "..\results\capreg\res_%BAS%.gdx"
$load meta
$gdxin
$label after
```

Further statement in the code introduce then meta data about the current run (loaded via the generated GAMS code in files a forreg.gms or fortran.gms), and in different data file. At the end of the programs, typically three statements will refer to meta data:

1. Storage of the meta data in the GDX data file together with the numerical values.

```
execute_unload "..\results\capreg\res_%BAS%.gdx" DATA2,RALL,COLS,ROWS,META;
```

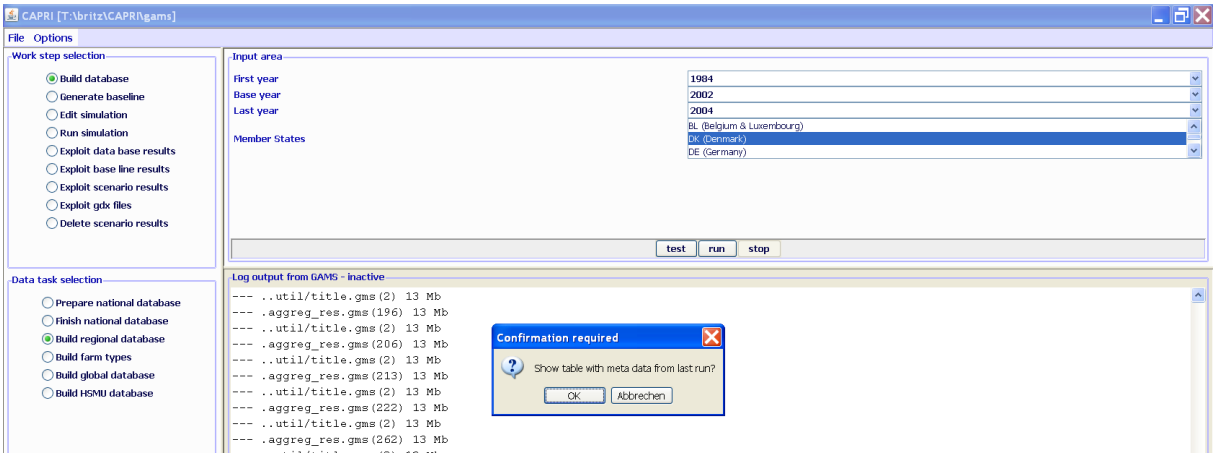
2. Temporary output of the meta data at compile time to allow the user to view meta data of the different data sets processed by the program before actual program execution. The temporary file will be deleted before program execution, and after the compile step, a dialogs allows the user to load the meta into the tabular viewer.

```
*
*  --- put meta data in temp. file to be loaded by GUI on demand
*
$GDXOUT meta.gdx
$UNLOAD META
$GDXOUT
```

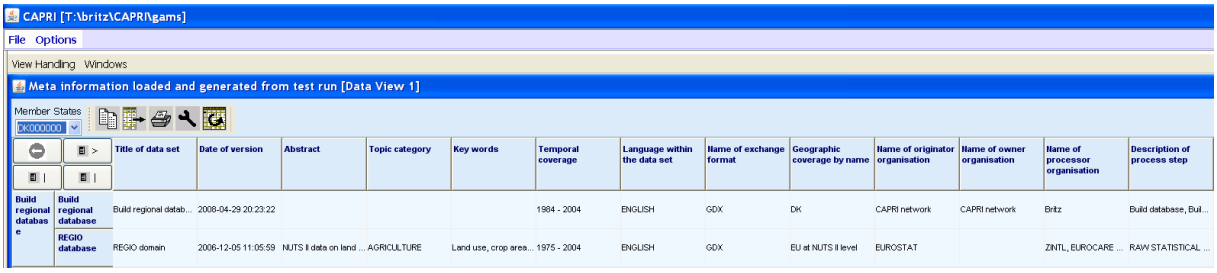
3. Output of the meta data in the temporary file to check the results directly after program execution, i.e. at run time, as above.

```
execute_unload "meta.gdx" META;
```

Check of meta data after a compile



At the end of the run, the JAVA GUI checks if the file “meta.gdx” had been generated. If that is the case, it queries the user if he wants to load the meta information. In the latter case, the meta data are loaded in the tabular viewer.

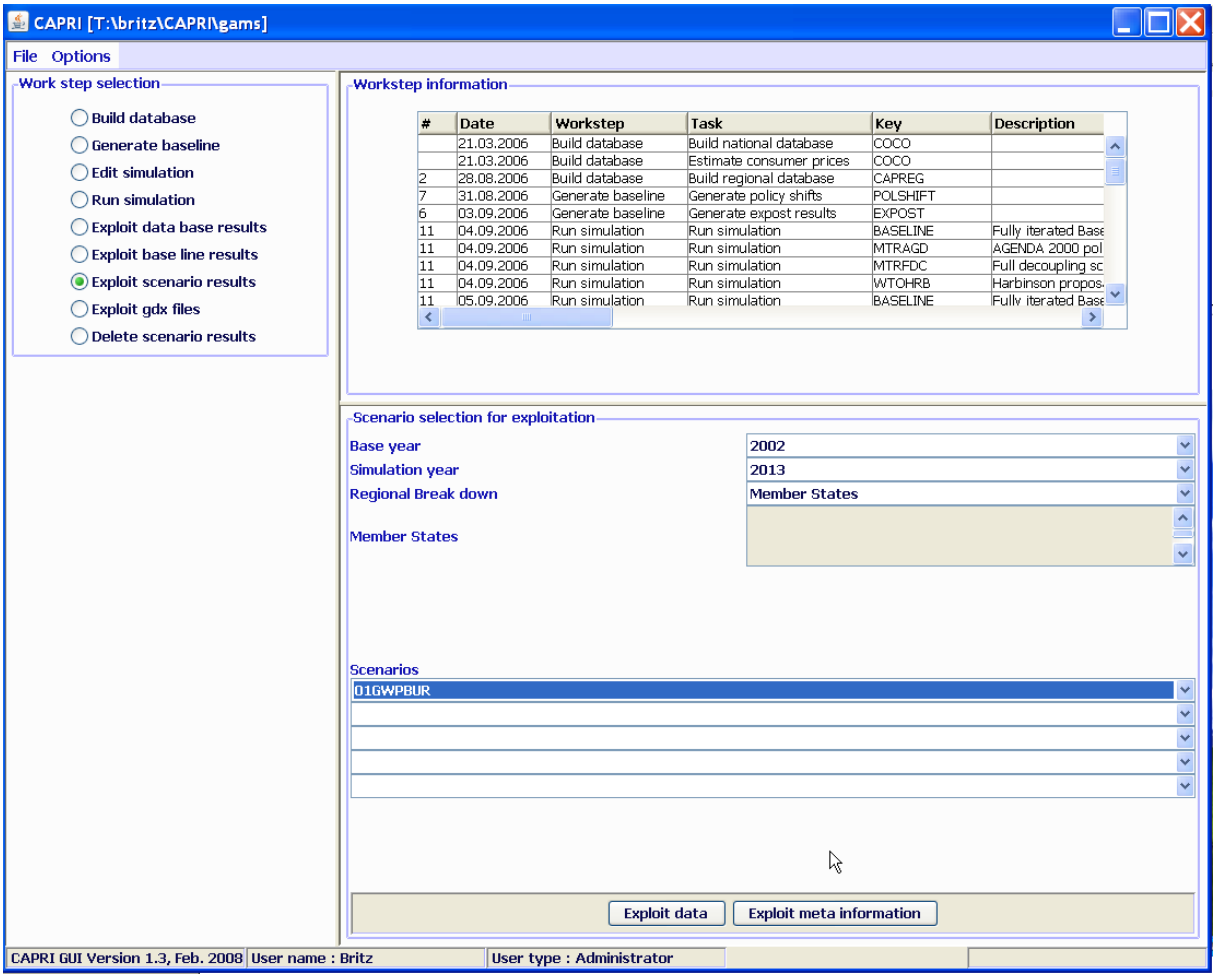


The user may now control in detail what data had been processed by the program. Currently, in CAPREG, solely the REGIO data are documented by meta data, further data sets as the Standard Gross Margins, the fertilizer data etc. should follow soon.

Exploitation of meta data in the GUI

The screenshot below shows the expanded GUI where now meta data can be queried as well. Once the user presses “Exploit meta information”, the long texts from the cross-set “META” are loaded from the GDX-files into a table and shown to the user, as seen below. The underlying java classes are a slightly expanded of the earlier version. As a by-product, CAPRI’s GDX viewer loads now also SETs into a tabular view.

The new GUI with exploitation possibilities for meta data



It may be astonishing to see that the meta information is four dimensional (member state, task, task, item), but that structure turned out as being useful. A three dimensional structure (member state, task, item) would in the table above report e.g. the generation date of the base year data from CAPREG used by the simulation run and the same information for the trends. It would however not reveal without loading in parallel meta information for the trends if the trends and the simulation run were based on the very same version of the CAPREG data. It was therefore judged necessary to use four dimensions.

Underlying is the ability of GAMS to copy also the long texts attached to a cross sets when copying it:

```
$gdxin "..\results\capreg\res_%BA$%.gdx"
$load meta
$gdxin
META(MSINCL,META_STEP,                  META_STEPS,META_ITEMS) = META(MSINCL,'Build regional database',META_STEPS,META_ITEMS);
META(MSINCL,'Build regional database',  META_STEPS,META_ITEMS) = 0;
```

The above code fragment from CAPTRD shows how the meta data are loaded from the GDX file with the CAPREG base year data, and then are copied over to the CAPTRD column, and

then being discarded. That allows consecutive steps using the trend projection to see the meta data attached to the base year data used by CAPTRD.

Currently, the meta data are integrated in the “tabular views” to allow the user to perform rapidly a few checks, e.g to compare the generate date and time of the different data sets used.

Introduction of code fragments into GDX files

Alexander Gocht has programmed a small Java program which stores the lines in a text file into a GDX file. That program can be executed by GAMS at compile time to document important parts of the code directly in the result set generated by the program. It uses the very same basic mechanism as above by storing strings as longtext description of GAMS sets. The facility is currently integrated in CAPMOD to report the content of “fortran.gms”, i.e. all the information passed from the GUI to the model, and the content of the included policy file. The following two statement show how (1) a possibly existing file named “%CURDIR%\temp.gdx” is deleted, and (2) the content of fortran.gms is stored in the set “SET_POL” in the file “%CUR_DIR%\temp.gdx”.

```
|
      execute    "%CURDIR%\F2GDX.jar new     SET_POL %CURDIR%\temp.gdx  %CURDIR%\pol_input\%result_type%.gms"
$IFI NOT %MODE%==SEAMCAP  execute  "%CURDIR%\F2GDX.jar append SET_POL %CURDIR%\temp.gdx  %CURDIR%\fortran.gms"
```

A similar statement adds then the content of the policy file:

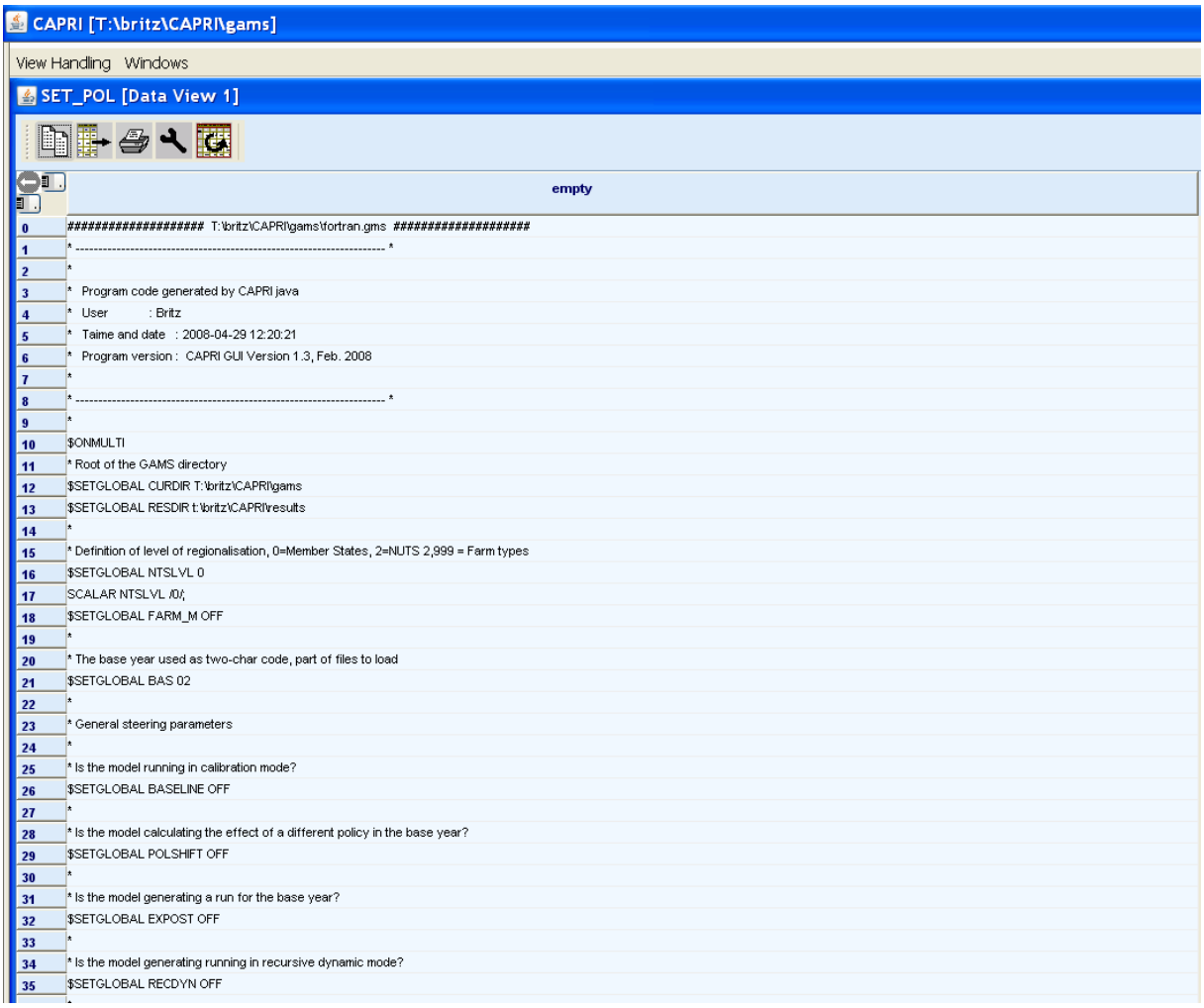
```
$IF NOT %MODE%==SEAMCAP  execute  "%CURDIR%\F2GDX.jar append SET_POL %CURDIR%\temp.gdx  %CURDIR%\pol_input\%result_type%.gms"
$IF NOT %MODE%==SEAMCAP  $include 'pol_input\%result_type%.gms'
```

At the end of the program, the content of the set is loaded from the GDX file and then outputted to the GDX file along with the numerical values and meta data:

```
  execute_load "%CURDIR%\temp.gdx" SET_POL;

$IF %EXPOST%==OFF  execute_unload "%resdir%\capmod\res_%NTSLVL%_%BAS%%SIM%%result_type%.gdx" DATAOUT,RALL,META,SET_POL;
$IF %EXPOST%==ON   execute_unload "%resdir%\capmod\res_%NTSLVL%_%BAS%%BAS%%result_type%.gdx" DATAOUT,RALL,META,SET_POL;
```

The content of the file can be opened either with the GAMSIDE or with the “exploit GDX” facility of the GUI:



That possibility clearly further on increases the transparency of the data as e.g. the details of the policy parameters passed in can be controlled.

Conclusions and look ahead

The GDX based passing of meta information is a step forward to a more transparent handling of meta data in CAPRI. It integration in the data files ensures that the information is not lost when the files are copied to the SVN-server, downloaded from the SVN-server or in another way shifted between directories or workstations. The integration in the usual exploitation tools make it easy to handle for users familiar with the GUI. However, understanding the information provided is another task which may require some training. As often, the system will receive its final layout once used in the real production chain.

The main advantage of the proposed solution is that the user can check for any result set per Member State meta data on the underlying data sets, and at least check if the very same input data had been used for different steps. The meta data will also only be available if the related GAMS program has successfully stored as well the numerical values. As such, the

implementation is less error prone and more transparent then the existing one based on a separate XML file which was never really put to work.

A further possibility based on JAVA code by Alexander Gocht is the possibility to dump the content of text file as e.g. selected GAMS code passages as sets into a GDX, which allows to store critical code fragments together with the numerical data.

The expansion of the data viewer to handle also strings instead of floats may lead to some surprises, and some further Java coding may be necessary to avoid run time errors. It must be decided to what extent e.g. export of the meta information to the clipboard or files is necessary.

Next possible steps could consists in adding such SET based meta information into major input data files from Eurostat, FAOSTAT .. which are subject to regular updates. That would enrich the meta information as the user could be informed e.g. when a statistical raw data had been downloaded, which time period was covered etc. Further steps as providing information regarding single items in the data base, as e.g. their numerical value changing along the production chain, require not only refactoring of the GAMS code, but will also increase memory needs. Pro and cons need therefore be discussed with the CAPRI network. Perhaps the most obvious candidate is CoCo, where such requests were already raised by major clients.

At current stage, the meta information handling was tested with CAPREG, CAPTRD and CAPMOD. Consent to expand it to COCO is reached. Some further coding is necessary to cover also the global data generation, the calculation of policy shifts and the calibration step.

Technically, it would also be possible to query information about the SVN release number for at least sub-directories, and possible also check for local modification , based on Java libraries. The added value of such a step needs to be discussed.

References

Hazeu, G., Verhoog, D., and Andersen, E. Metadata of environmental, farming system, socio-economic and global data selected to be implemented in the knowledge base. SEAMLESS deliverables PD 4.3.1., PD 4.4.1, PD 4.5.1., PD 4.6.1, February 2006