The Updated BODC Parameter Dictionary

Update History

Initial draft	RKL	21/10/2003
Revised draft	RKL	09/11/2003
Test configuration	RKL	14/11/2003
Tested and debugged	RKL	20/11/2003

The Case for Change

The BODC parameter dictionary is based on a relational structure that was defined in 1987 based upon a design concept formulated in the 1970s to deal with a very small number of parameters (10s at most). During the evolution of the dictionary to cover more and more oceanographic parameters the premises of this design have been stretched beyond recognition.

The original parameter dictionary consisted of a four-byte parameter code, followed by a 2-byte parameter subgroup code and a 2byte parameter discriminator (for cases where there were multiple occurrences of the same parameter in a file). The definition of a 'parameter subgroup' has become somewhat fuzzy over time through shoe-horning chemical and biological parameter descriptions into 8-byte codes. This also caused the usage of bytes 7 and 8 to be relaxed for codes in CRP tables, but these remained as discriminators for PXF parameters due to the design of the Fortran Transfer System. The new Matlab-based Transfer System places no constraints on the parameter code structure leaving us free to use the 8 bytes in any way we wish.

The concept of a structured code is therefore steadily eroding. There is a strong case to take this forward and abandon the concept of structured codes altogether. This is because the structure implements a hierarchical classification into the label used for marking up parameters within data files and tables. Consequently, any reclassification of parameters (or fixing errors in the assignment of CPMREF) can only be achieved by changing codes and hence modification to existing data mark-up. This is possible, but costly for BODC, and is unlikely to endear us to external dictionary users. Note that I am not proposing that we abandon ZUPM (at least not until we have a better classification mechanism). The proposal is simply that the ZUPM foreign key in ZUSG (CPMREF) ceases to specify bytes 1-4 of CPMUSG. Indeed, I would go further than this and suggest that this relationship should be 'encouraged but not mandatory' to minimise the confusion to those marking up the data.

The decoupling of CPMREF and CPMUNT is the first stage towards converting ZUPM into a pure hierarchical classification of the dictionary entries held in ZUSG. This process could be completed by transferring several fields normalised into ZUPM (FABSNT, FPMINM, FPMAXM and CPMUNT) into ZUSG. This will solve problems caused by parameters having 'subgroups' with expected values differing by orders of magnitude (e.g. dissolved and particulate trace metals). We can also take the opportunity to incorporate the temporary fix table (ZUSG_SIG) into ZUSG.

The most pressing case for change now is that EnParDis plus other initiatives like SGXML will dramatically raise the profile of the BODC parameter dictionary. It would seem prudent to implement any radical changes before this happens.

Costs of Change

Significant changes to the structure of ZUPM and ZUSG do not come cheap. Whilst the changes to the Oracle tables are relatively easy to implement, usage of parameter codes are endemic throughout our software infrastructure. Of particular concern are instances where the assumption that the first four bytes of the code are significant has been incorporated into the software design. Upgrading the software to remove such assumptions will require more extensive work than simply modifying the interface routines and regenerating the binaries. Note that this engineering will not be confined to our FORTRAN systems and developers in all technologies will need to carefully consider the implications for their software responsibilities.

Scope of Change

The changes eluded to so far concern giving parameter classification into groups total flexibility and the elimination of a 'quick fix' (ZUSG_SIG). The classification change between ZUSG and ZUPM is far from revolutionary. The straightforward one-to-many relationship between parameter codes and groups of parameter codes remains. All that has changed is that the foreign key is no longer embedded in the parameter code.

However, it is felt that a more radical change is required for the higher level classification of parameter groups. External forces, especially the fact that parameter grouping is an extremely emotive subject, are driving us towards a situation where we have to offer multiple, even user-definable, classifications. The adoption of DIFs by NERC DataGrid for data discovery means that one of the classifications we need to support is the GCMD Parameter Valids list. These are constructed in such a way that a simple one-to-many mapping is impossible and we need to implement a many-to-many classification. As we are doing this for GCMD it would seem sensible to establish the same structures for our other mapping (SEASEARCH agreed parameter groupings) and for any further classifications developed in the future. Keeping the structures internally consistent makes switching classification a simple matter of table name substitution.

In practice, implementation of the many-to-many relationship means replacing table ZUCT by two tables one containing a list of the classification categories and the other a two-key link table. Note that research is already underway to develop other types of parameter classification that bypass the fixed one-to-many grouping implemented through ZUPM.

As change is associated with significant costs, it seemed sensible to critically analyse the dictionary we have and incorporate changes to correct any weaknesses identified at the same time. Areas of weakness were identified as:

- The clarity of some parameter descriptions has been compromised due to space limitations of ZUSG fields, particularly CSGMTH.
- Native SQL is being used to maintain the dictionary due to concerns about accidental field value changes through an Access interface.
- There is no mechanism to support maintenance of dictionary entries for the benefit of external users that we consider go against what we consider 'best practice' for usage within BODC. For example, support of parameters stored in multiple units.
- The units code table (ZUNT) has become very scrappy, partially due to field size constraints. Additional information, such as a comments field, would help eliminate errors in the future.
- The naming of tables and columns using cryptic coding conventions is somewhat anachronistic and inappropriate for a product designed for external consumption.

The changes described below address these issues.

Dictionary Storage and Maintenance Environment

The dictionary tables will be stored under the Oracle user id bodcdb, which will have full privileges to create, modify and therefore destroy dictionary table content. Safe dictionary maintenance through an Access front end therefore cannot be set up for a user logged in as bodcdb. Configuring bodc as the maintenance id is also not what we require as policy is to restrict dictionary maintenance to designated individuals. Consequently, a new restricted Oracle id (bodcpm) will be set up specifically parameter dictionary maintenance.

Changes to ZUCT

Table ZUCT will be replaced by a series of table pairs, one pair for each classification supported. The names of each table pair will take the form:

xxxx_category
xxxx_bodc_link

where xxxx is the name of the classification, but the number of 'x' characters has no significance (eg seasearch_category is a valid table name). Two classifications will be implemented initially called:

SEASEARCH	SEASEARCH agreed parameter groups (APGs)
GCMD	NASA Global Change Master Directory Parameter Valids

The structure of the 'category' tables is:

code	VARCHAR2(4)	NOT NULL	Reference code for the classification category (primary key)
title	VARCHAR2(200)	NOT NULL	Classification category title
docref	NUMBER(8)	NOT NULL	Reference number of an XHTML document in the BODC documentation system describing the category
record_lock	VARCHAR2(1)		Record modification lock (locked if null)
created	DATE	NOT NULL	Date/time the record was created
modified	DATE	NOT NULL	Date/time the record was last modified

The structure of the 'link' tables is:

category_code	VARCHAR2(4)	NOT NULL	Reference code for the classification category
group_code	VARCHAR2(4)	NOT NULL	Reference code for the BODC parameter grouping

Audit trail tables, xxxx_category_deleted and xxxx_category_modified, will be maintained. A view is required for the subset of records that have been unlocked for modification, 'xxxx_category_unlocked', defined as:

```
select * from xxxx_category where record_lock is not null
```

No views are required on the link tables.

Triggers

A trigger should be set up to store the record in xxxx_category_modified when record_lock is changed from NULL to NOT NULL.

A trigger should be set up to set modified to SYSDATE when record lock is changes from NOT NULL to NULL.

An 'on delete' trigger should be set up to store the record and the date/time of deletion.

Permissions

All BODC Oracle ids (bodc, odb, sdb, pmlamt, bodcweb, omex, bodcpm and bodcconv) need to have select permission on xxxx_category, xxxx_category_deleted, xxxx_category_modified and xxxx_link.

In addition, bodcpm needs the following permissions:

- Select permission on xxxx_category_unlocked
- Insert permission on xxxx_category, xxxx_category_deleted, xxxx_category_modified and xxxx_link
- Delete permission on xxxx_category_unlocked and xxxx_link
- Update permission on the following fields
    - xxxx_category.record_lock
    - xxxx_category _unlocked.modified
    - all xxxx_category _unlocked fields except code and created

Constraints

xxxx_category.code	Primary key
xxxx_bodc_link	Primary key comprising both fields
xxxx_category.title	May not include any double quote (") characters
xxxx_category.docref	Foreign key to be defined in zond.indref
xxxx_bodc_link.category_code	Foreign key to be defined in xxxx_category.code
xxxx_bodc_link.group_code	Foreign key to be defined in parameter_group.code

Changes to ZUNT

Table ZUNT is replaced by table 'units', which has the following structure:

code	VARCHAR2(4)	NOT NULL	Reference code for the parameter unit
short_title	VARCHAR2(20)	NOT NULL	Abbreviated title for the parameter unit
full_title	VARCHAR2(100)	NOT NULL	Full title for the parameter unit
comments	VARCHAR2(4000)		Additional information on parameter usage
record_lock	VARCHAR2(1)		Record modification lock (locked if null)
created	DATE	NOT NULL	Date/time the record was created
modified	DATE		Date/time the record was last modified

Audit trail tables, units_deleted and units_modified will be maintained.

A view is required for the subset of records that have been unlocked for modification, 'units_unlocked', defined as:

```
select * from units where record_lock is not null
```

Triggers

A trigger should be set up to store the record in units_modified when record_lock is changed from NULL to NOT NULL.

A trigger should be set up to set modified to SYSDATE where record_lock is changes from NOT NULL to NULL.

An 'on delete' trigger should be set up to store the record and the date/time of deletion.

Permissions

All BODC Oracle ids (bodc, odb, sdb, pmlamt, bodcweb, omex, bodcpm and bodcconv) need to have select permission on units, units_deleted and units_modified.

In addition, bodcpm needs the following permissions:

- Select permission on units_unlocked
- Insert permission on units, units_deleted and units_modified
- Delete permission on units_unlocked
- Update permission on the following fields
    - units.record_lock
    - units_unlocked.modified
    - all units_unlocked fields except code and created

Constraints

code	Primary key
short_title	May not include any double quote (") characters
full_title	May not include any double quote (") characters
comments	May not include any double quote (") characters

Changes to ZUSG

Table ZUSG is replaced by table 'parameter', incorporating the following structural changes:

- Fields CSGREF and CDSREF are dropped.
- The fields FABSNT, FPMINM, FPMAXM and CPMUNT, formerly in ZUPM, are added.
- A field sig_fig is added for the number of significant digits to be displayed.
- CSGABB is replaced by 'short_title' and increased in size to 50 bytes.
- CSGFUL is replaced by 'full_title' and increased in size to 200 bytes.
- CSGMTH is replaced by 'definition' and increased in size to 4000 bytes.
- Field ISGREF is deleted (functionality replaced by the uprated 'definition' field).
- A field record_lock is added to allow 'safe' editing through Access forms
- A field bodc_legal is added to specify a subset of codes that may be used within BODC.
- CILOAD is replaced by created, a field of type date.

These are manifested in the following table definition:

code	VARCHAR2(8)	NOT NULL	Parameter code (primary key)
group_code	VARCHAR2(4)	NOT NULL	Parameter group code (foreign key)
unit_code	VARCHAR2(4)	NOT NULL	Parameter unit code (foreign key)
dummy_val	NUMBER	NOT NULL	Parameter dummy value (i.e. absent data) code
min_permiss_val	NUMBER	NOT NULL	Minimum valid value for parameter
max_permiss_val	NUMBER	NOT NULL	Maximum valid value for parameter
before_dp	NUMBER(2)	NOT NULL	Number of figures (excluding sign) to be output before the decimal place
after_dp	NUMBER(1)	NOT NULL	Number of figures to be output after the decimal place
sig_fig	NUMBER(2)		Number of significant figures to be output. Setting this field not null causes BODC presentation software to display in scientific notation.
short_title	VARCHAR2(50)	NOT NULL	Abbreviated parameter title
full_title	VARCHAR2(200)	NOT NULL	Full parameter title
definition	VARCHAR2(4000)	NOT NULL	Parameter definition, including descriptive information and usage notes.
record_lock	VARCHAR2(1)		Record modification lock (locked if null)
bodc_legal	VARCHAR2(1)		Parameter code valid for BODC usage (valid if null)
created	DATE	NOT NULL	Date/time record was created
modified	DATE		Date/time record was last modified

It is proposed that this new table be called parameter with three views:

The parameter dictionary as to be used within BODC, 'parameter_bodc' defined as:

```
select * from parameter where bodc_legal is null
```

The subset of records that have been unlocked, 'parameter_unlocked' defined as:

```
select * from parameter where record_lock is not null
```

The subset of parameters used only into the CRP databases that are unlocked for modification, 'parameter_crp', defined as:

```
select * from parameter where bodc_legal is null and code not in
(select cpmusg from bodcdb.zypsu) and record_lock is not null
```

This view will be used to restrict dummy value code changes to parameters that are not included in the QXF file stock to prevent invalidation of QXF files through changes to the database.

The two existing 'audit trail' tables (ZUSG_MODIFIED and ZUSG_DELETED) will be maintained under the names 'parameter_modified' and 'parameter_deleted', and their contents will be re-engineered to the new structure.

Triggers

A trigger should be set up to store the record in parameter_modified when record_lock is changed from NULL to NOT NULL.

A trigger should be set up to set modified to SYSDATE when record_lock is changes from NOT NULL to NULL.

An 'on delete' trigger should be set up to store the record and the date/time of deletion.

A 'before insert or update' trigger to ensure that before_dp is big enough to accommodate dummy_val and min_permiss_val and max_permiss_val.

Permissions

All BODC Oracle ids (bodc, odb, sdb, pmlamt, bodcweb, omex, bodcpm and bodcconv) need to have select permission on parameter_bodc, ,parameter_deleted and parameter_modified.

All BODC Oracle ids need to have references permission on parameter_bodc.

In addition, bodcpm needs the following permissions:

- Select permission on parameter, parameter_unlocked and parameter_crp
- Insert permission on parameter, parameter_deleted and parameter_modified
- Delete permission on parameter_unlocked
- Update permission on the following fields
    - parameter.record_lock
    - parameter_unlocked.modified
    - all parameter_unlocked fields except code, dummy_val and created
    - parameter_crp.dummy_val

These permissions provide the record locking mechanism plus a mechanism to prevent absent data values used in the NODB file stock being modified without exceptional procedures being invoked.

Constraints

The following constraints need to be placed on this table:

code	Primary key
unit_code	Foreign key to be defined in units.code
group_code	Foreign key to be defined in parameter_group.code
dummy_val	Must be a negative number <min_permiss_val
min_permiss_val	Must be >dummy_val and <=max_permiss_val
max_permiss_val	Must be >= min_permiss_val
short_title	May not include any double quote (") characters
full_title	May not include any double quote (") characters
definition	May not include any double quote (") characters

Note that NOT NULL constraints have not been included as it is assumed that they will be implemented through the table creation command.

Changes to ZUPM

The table ZUPM will be replaced by the table 'parameter_group' incorporating the following structural changes:

- Fields CPMUNT, FABSNT, FPMINM, FPMAXM, CPMCAT and CINVER are dropped.
- CPMABB is defined as 'Abbreviated parameter group title' and increased in size to 50 bytes.
- CPMFUL is defined as 'Parameter group title' and increased in size to 200 bytes.
- CPMMTH is added as a large (4000-byte) text field for the definition of the grouping.
- A field RECORD_LOCK is added to allow 'safe' editing through Access forms.
- A field BODC_LEGAL is added to specify a subset of parameter groups that may be used within BODC.
- CILOAD is replaced by TPMLOAD, a field of type date.

These are manifested in the following table definition:

code	VARCHAR2(4)	NOT NULL	Parameter group code (primary key)
short_title	VARCHAR2(50)	NOT NULL	Parameter group abbreviated title
full_title	VARCHAR2(200)	NOT NULL	Parameter group title
definition	VARCHAR2(4000)		Parameter group definition, including descriptive information and usage notes.
record_lock	VARCHAR2(1)		Record modification lock (locked if null)
bodc_legal	VARCHAR2(1)		Parameter group valid for BODC usage (valid if null)
created	DATE	NOT NULL	Date/time record was created
modified	DATE		Date/time record was last modified

This new table will have two views:

parameter_group_bodc defined as:

```
select * from parameter_group where bodc_legal is null
```

parameter_group_unlocked defined as:

```
select * from parameter_group where record_lock is not null
```

The two existing 'audit trail' tables (ZUPM_MODIFIED and ZUPM_DELETED) will be maintained under the names parameter_group_modified and parameter_group_deleted and their content re-engineered to the new structure.

Triggers

A trigger should be set up to store the record in parameter_group_modified when record_lock is changed from NOT NULL to NULL.

A trigger should be set up to set modified to SYSDATE when record_lock is changes from NOT NULL to NULL.

An 'on delete' trigger should be set up to store the record and the date/time of deletion.

Permissions

All BODC Oracle ids (bodc, odb, sdb, pmlamt, bodcweb, omex, bodcpm and bodcconv) need to have select permission on parameter_group_bodc, parameter_group_deleted and parameter_group_modified.

In addition, bodcpm needs the following permissions:

- Select permission on parameter_group, parameter_group_bodc and parameter_group_unlocked
- Insert permissions on parameter_group, parameter_group_deleted and parameter_group_modified
- Delete permissions on parameter_group_unlocked
- Update permission on the following fields
    - parameter_group.record_lock
    - parameter_group_unlocked.modified
    - all parameter_group_unlocked fields except code and created

This provides the record locking mechanism.

Constraints

The following constraints need to be placed on this table:

code	Primary key
short_title	May not include any double quote (“) characters
full_title	May not include any double quote (“) characters
definition	May not include any double quote (“) characters

Note that NOT NULL constraints have not been included as it is assumed that they will be implemented through the table creation command.

Implementation Plan

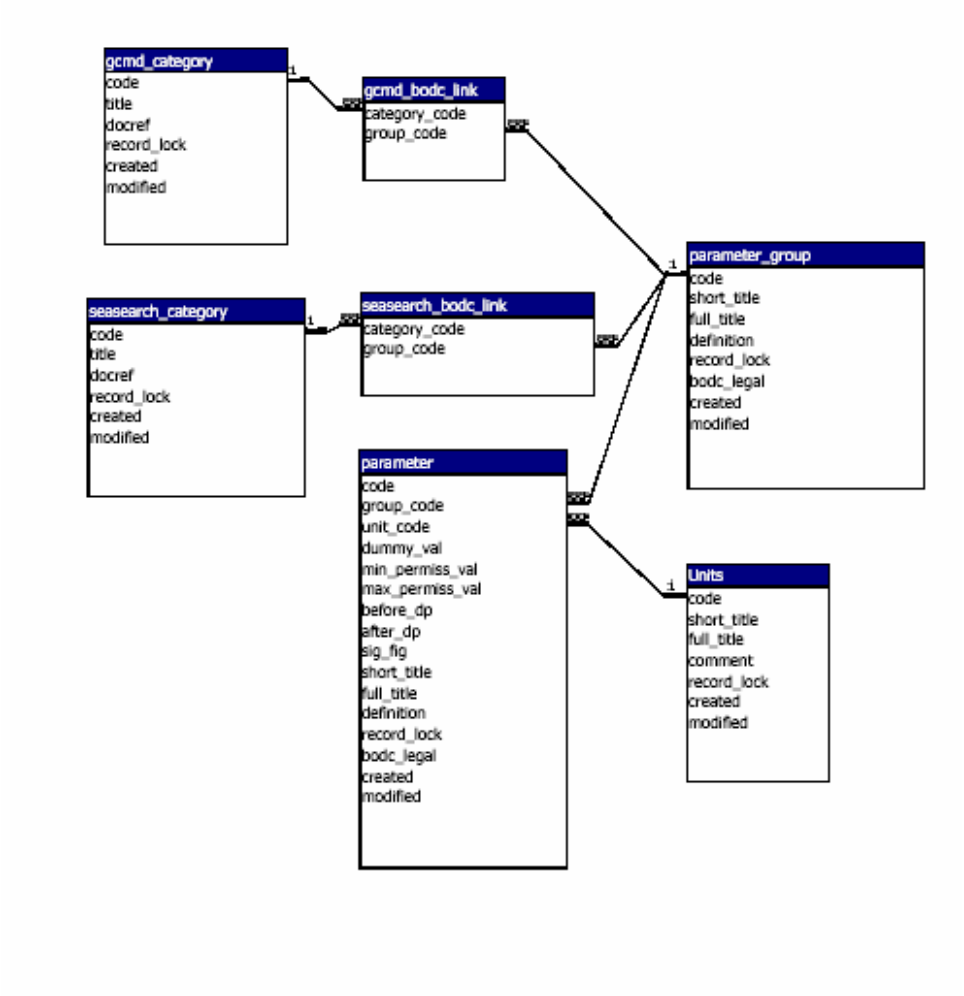
The following plan is proposed for implementation of these changes:

- Announce the changes to known external users, NERC DataGrid, Marine XML, SEASEARCH and SGXML.
- Set up the table creation start files
- Create the new dictionary tables
- Freeze the existing ZUPM and ZUSG tables
- Initial population of the new tables and initiation of maintenance on the new structures
- Commence weekly export of new structures to dictionary ftp site
- Modify foreign key constraints on the dictionary to point to parameter_bodc.code instead of ZUSG.CPMUSG
- Commence software conversion
- Fields of the NODB document update
- Remove synonyms and permissions to old dictionary tables
- Archive off CSV exports of the final frozen version of ZUCT, ZUNT, ZUPM and ZUSG
- Delete old dictionary tables from bodcdb

I currently plan to have the new tables live by the end of November and it would be preferable to have all the associated software maintenance completed by the end of January 2004. If I’ve read it right our application software should be able to function on the frozen dictionary for a couple of

months at least without any operational consequences. However, the longer we leave it the more likely we are to hit problems.

The primary responsibility for this work falls to Steve (Matlab/Java systems), Ray (Delphi systems), Laura/Mary (Access interface) and Laura/me (Oracle setup, FORTRAN systems and documentation). However, significant change to one of our underpinning information resources is bound to have far reaching consequences. **Anybody in the group who has written anything that accesses Oracle should check their code to see if there is any maintenance required.**



Appendix: Access Relationship Diagram of the new Parameter Dictionary