EXTRACTION DES DONNÉES AUX FORMATS BUFR, NETCDF & HDF5 POUR ARPEGE/ALADIN/AROME.

VERSION FRANÇAISE / FRENCH VERSION

V. 1.2.6 (POUR CENTOS V. 6.XX)

TABLE DES MATIÈRES

1 Avant de commencer....5
1.1 Abréviations utilisées....5
1.2 Conventions typographiques....5
2 Introduction....5
3 Historique....5
4 Utilisation du script alim.awk....6
4.1 Choisir l'environnement d'exécution désiré....7
4.2 Éléments en entrée....8
4.3 Fichiers produits....8
4.3.1 Fichiers binaires BUFR & NETCDF (/HDF5)....8
4.3.2 Fichiers refdata....8
4.3.3 Fichiers batormap....9
4.3.4 Fichier ouloutput....9
4.4 Le cas particulier des NETCDF/HDF5....10
5 le FDEO....10
5.1 Les types de phrases reconnues....10
5.2 Règles d'écritures des phrases....10
5.3 Listes des balises autorisées....11
5.3.1 <EXTRACTION> </EXTRACTION>....11
5.3.2 <HEADING COLUMN> </HEADING COLUMN>....11
5.3.3 <DATA COLUMN> </DATA COLUMN>....11
5.3.4 <BATORMAP> </BATORMAP>....12
5.4 Liste des clés autorisées....12
5.4.1 OBS TYPE....13
5.4.2 UPPER LATITUDE....13
5.4.3 LOWER LATITUDE....13
5.4.4 WESTER LONGITUDE....13
5.4.5 EASTER LONGITUDE....14
5.4.6 PRECISE UPPER LATITUDE....14
5.4.7 PRECISE LOWER LATITUDE....14
5.4.8 PRECISE WESTER LONGITUDE....15
5.4.9 PRECISE EASTER LONGITUDE....15
5.4.10 SQL WHERE....15
5.4.11 PRECISE OBS DATE....15
5.4.12 DELTA BEFORE....16
5.4.13 BEFORE EXTENDED....16
5.4.14 DELTA AFTER....16
5.4.15 CUTOFF....17
5.4.16 BINARY OUTPUT....17
5.4.17 BINARY FILENAME....17
5.4.18 REFDATA....17
5.4.19 TARGET BASE....17
5.4.20 SENSOR....18
5.4.21 MIN INDIC....18
5.4.22 MAX INDIC....18
5.4.23 CONCATENATION....18
5.4.24 SQL FROM....18
5.4.25 SQL ORDER BY....19
5.4.26 RIGHTS....19
5.4.27 LAST ARRIVED....19
5.4.28 ARCHIVED DATA....19

5.4.29 ASCII OUTPUT....19
5.4.30 ASCII COMMENTED....19
5.4.31 ASCII FILENAME....20
5.4.32 SEPARATOR....20
5.4.33 PRINT MISSING VAL....20
5.4.34 HEADING COL NAME....20
5.4.35 HEADING COL FMT....20
5.4.36 DATA COL NAME....21
5.4.37 DATA COL FMT....21
5.4.38 DATA COL MISSING....21

Annexe – 1 : date pivot & metronome (extrait du vade-mecum DSI)

1 Date-pivot et Profil de répétition....25
2 Questions de base....25
3 Le développement....25
3.1 Un exemple....26
4 Expression de la date pivot....26
5 Expression de date d'utilisation de ressource....26

Annexe – 2 : les outils disponibles

1 Fichier alim.awk & directives d'extraction....29
2 Extractions via Olive....29
3 Outils complémentaires....29
4 Outils de manipulations des fichiers BUFR....29

Annexe – 3 : quelques exemples

1 Extraction globale simple....31
2 Extraction globale avec concaténation et clause where....31
3 Extractions avec heure précise et clauses where & order by....33
4 Extraction à domaine limité avec concaténation....35
5 Extraction de données au format NETCDF (exemple valable pour HDF5)....36
6 Extraction de données MTVZA au format HDF5....37
7 Demande d'extraction en ASCII – mode recherche....38

1 Avant de commencer

1.1 Abréviations utilisées

- FDEO : Fichier de Description des Extractions des Observations.
- BDMO : Base de Données Météorologiques – Observations.
- ASCII : American Standard Code for Information Interchange.
- XML : eXtensible Markup Language.
- HTML : HyperText Markup Language.
- DSI : Direction des Systèmes d'Informations.
- ODB : Observational DataBase.

1.2 Conventions typographiques

- Les noms de fichiers, de tâches ou de programmes sont imprimée en **gras**.
- Les types de blocs ou de phrases utilisés dans le **FDEO** sont imprimés en *italique*.
- Les exemples de code sont imprimés en utilisant la police `Courier New`.
- Dans les exemples de code, les termes apparaissant entre crochets sont facultatifs.
- De même les termes imprimés en *`Courier New italique`* doivent être remplacés par leur valeur.
- Dans les exemples, si nécessaire, le caractère espace est matérialisé par « ▫ ».

2 Introduction

Ce document a pour objectif de donner les éléments nécessaires à l'extraction des données (BUFR, NETCDF et HDF5) de la BDMO en utilisant les outils disponibles sur SOPRANO ; ces outils sont ceux utilisés dans le cadre de la chaîne opérationnelle et sous Olive pour alimenter les modèles atmosphériques ARPEGE, ALADIN, et AROME, utilisés à Météo-France.
L'utilisation des binaires **oulan** et **bator**, ainsi que de l'utilitaire **dap3** ne sera pas abordée ici.

3 Historique

Version 1.2.6 (16/06/2020) :
- bugfix affectant les fichiers **batormap** lorsque des fichiers d'observations au format BUFR possèdent une racine commune dans leur extension (ex. **BUFR.amdar** et **BUFR.amdaromm**)

Version 1.2.5 (16/01/2018) :
- bugfix dans le traitement de la clef `BEFORE EXTENDED` si la fenêtre d'assimilation est différente de [-3h : +3h[.

Version 1.2.4 (31/05/2017) :
- ajout de la clef `BEFORE EXTENDED` dans le **FDEO**.

Version 1.2.3 (25/07/2016) :
- adaptation de certaines requêtes à la syntaxe PostgreSQL.

Version 1.2.2 (24/05/2016) :
- adaptation de la documentation pour le cycle CY42_op1 et suivants.
- ajout des extractions au format HDF5.

Version 1.2.1 (07/10/2015) :
- prise en charge d'une seconde syntaxe pour la clef `REFDATA`,

- adaptation aux extractions particulières au format NETCDF du modèle Arome.

Version 1.2.0 (24/08/2015) :
- prise en charge de l'extraction des fichiers au format NETCDF contenus dans la BDMO,
- nettoyage du code.

Version 1.1.4 (05/08/2015) :
- suppression de l'utilisation de la variable d'environnement `SWAP_BDM_ARCH_ENV`.

Version 1.1.3 (10/04/2015) :
- changement de la valeur par défaut de `MIN INDIC` pour prendre en compte certains messages qui comporte une erreur de codage dans l'indicatif OMM.

Version 1.1.2 (12/08/2008) :
- suppression des spécificités DIAPASON et fusion entre les versions opérationnelles et Olive.

Version 1.1.1 (18/03/2008) :
- adaptation à l'environnement SOPRANO.

Version 1.1.0 (28/02/2008) :
- ajout de la clef `PRECISE OBS DATE` dans le **FDEO**,
- obligation de renseigner les clés `SENSOR` et `TARGET BASE` de la balise `<BATORMAP>`.

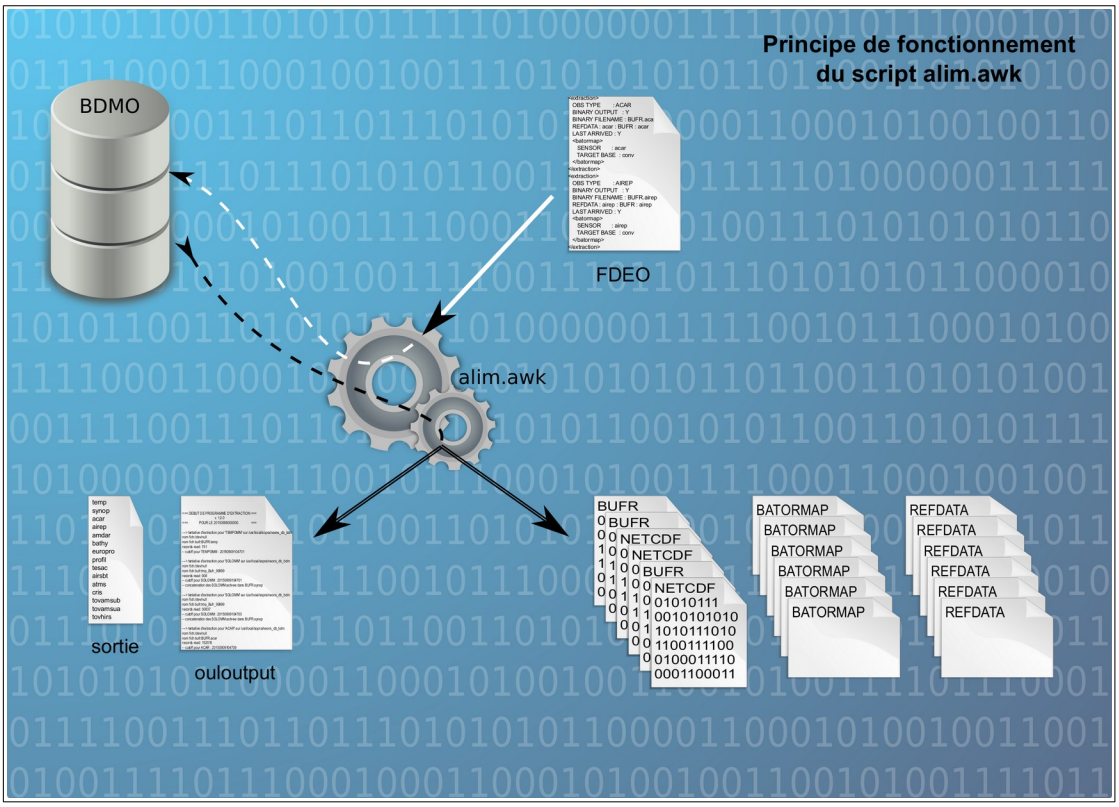
Version 1.0.5 (24/10/07) :
- prise en charge de l'extraction et de la décompression des fichiers de données radar,
- création d'une version spécifique du script pour Olive pour gérer les bases sous DIAPASON et SOPRANO.

Version 1.0.4 (12/04/2007) :
- ajout de la balise `<BATORMAP>` et de ses 2 clés `SENSOR` et `TARGET BASE` pour générer un fichier batormap,
- ajout de la clef `CUTOFF` pour simuler un cutoff opérationnel,
- ajout des clés `PRECISE UPPER LATITUDE`, `PRECISE LOWER LATITUDE`, `PRECISE WESTER LONGITUDE` et `PRECISE EASTER LONGITUDE` dans le **FDEO**,
- la sortie dans le fichier **ouloutput** est améliorée.

Version 1.0.3 (11/05/2006) :
- ajout de la clef `CONCATENATION`,
- utilisation de la variable d'environnement `EXTR_ENV` pour contrôler l'environnement d'exécution du script.

Version 1.0.0 (14/09/2005) :
- première version du script.

4 Utilisation du script alim.awk



4.1 Choisir l'environnement d'exécution désiré

Par défaut, le script awk est configuré pour une utilisation similaire à celle faite dans le cadre opérationnel (BDMO en ligne et environnement OPERATIONNEL). Toutefois, il est possible d'adapter son comportement à ce que l'on souhaite en utilisant les deux variables d'environnement `EXTR_ENV` et `DB_FILE_BDM`. Ci-dessous un tableau donnant les différents comportements possibles en fonction des valeurs prises par ces deux variables.

Base demandée	**$DB_FILE_BDM =**	**SI $EXTR_ENV = RECHERCHE**	**SI $EXTR_ENV = OPERATIONNEL**
En ligne	`/usr/local/sopra/neons_db_bdm`	`last_arrived = N`	`last_arrived = Y`
Archive	`/usr/local/sopra/neons_db_bdm.archi`		
Intégration	`/usr/local/sopra/neons_db_bdm.intgr`		

Les BDMO en ligne et d'intégration peuvent être interrogées pour extraire des données récentes (rétention jusqu'à 5 jours). Pour toute extraction au-delà de cette durée de rétention, seule la BDMO archive (qui ne contient pas les observations spécifiques à la BDMO d'intégration) peut être interrogée.

Lorsque `LAST_ARRIVED = N`, tous les éventuels avenants aux messages sont extraits.

*Si la clef `LAST ARRIVED` est présente dans un bloc d'extraction du **FDEO**, sa valeur prédomine, pour ce bloc uniquement, celle définie par l'environnement.*

4.2 Éléments en entrée

Une fois l'environnement choisi, il est nécessaire de fixer la date pivot, à l'aide de la variable d'environnement `DMT_DATE_PIVOT`,(cf. Annexe – 1 Page 25) et de fournir un Fichier de Description des Extractions des Observations, ou **FDEO** (pour la syntaxe de ce fichier cf. 5). Ci-dessous un exemple de script.

```
#!/bin/bash

# petit script pour lancer une extraction en mode operationnel,
# en utilisant la BDMO archive.
# syntaxe :     extract nom_namelist date_pivot

if test $# -lt 2
  then echo "syntaxe incorrecte"
    echo " extract nom_namelist date_pivot"
  else
    export DMT_DATE_PIVOT=$2
    export EXTR_ENV=OPERATIONNEL
    export DB_FILE_BDM=/usr/local/sopra/neons_db_bdm.archi
    awk -f alim.awk $1 | tee ouloutput
fi
```

4.3 Fichiers produits

Selon le contenu du **FDEO**, sont produits un ou plusieurs fichiers binaires au format BUFR (ou NETCDF), chacun d'entre eux étant accompagné de 2 fichiers ASCII, un de type **refdata** et l'autre de type **batormap**.
Sont également produits 2 fichiers de contrôle : **ouloutput** et **SORTIE**.

4.3.1 Fichiers binaires BUFR & NETCDF (/HDF5)

Ces fichiers ont pour vocation d'être transférés vers le supercalculateur pour y être décodés (actuellement par le binaire **Bator**). Les données qu'ils contiennent sont alors stockés au format ODB.

4.3.2 Fichiers refdata

À chaque fichier binaire extrait on associe un fichier ASCII appelé **refdata**, dont le but est de décrire succinctement les données contenues dans le fichier BUFR ou NETCDF pour en permettre le décodage par **Bator**.

*Les fichiers **refdata** générés par le script awk ne sont plus utilisés par les tâches **Bator** des chaînes opérationnelles de Météo-France à partir du CY42_op1. On leurs préfère des fichiers **batormap**.*

Il est composé d'un enregistrement, possédant 5 champs séparés par un espace. Sa structure est la suivante :

- l'extension du fichier binaire (et celle du refdata lui-même), en 8 caractères,
- le format du fichier binaire, en 8 caractères – pour le moment seuls les formats BUFR NETCDF et HDF5 sont gérés (OBSOUL et GRIB le sont également via **Oulan** et **dap3**),
- le type de données ou de capteur, en 16 caractères,
- la date pivot sous la forme YYYYMMDD,
- le réseau extrait sous la forme HH.

Ci dessous plusieurs exemples de fichiers **refdata** :

```
tovamsua▫BUFR▫▫▫▫▫amsua▫▫▫▫▫▫▫▫▫▫▫20150908•00
europro▫▫BUFR▫▫▫▫▫europro▫▫▫▫▫▫▫▫▫20150908•00
sev000▫▫▫NETCDF▫▫▫seviri▫▫▫▫▫▫▫▫▫▫20150908•00
```

4.3.3 Fichiers batormap

A chaque fichier binaire extrait on associe également un fichier ASCII appelé **batormap**, dont le but est de placer les données contenues dans le fichier BUFR, NETCDF ou HDF5, dans une base ECMA particulière (ODB).

*Les fichiers **batormap** générés par le script awk sont utilisés, après traitement, par les chaînes opérationnelles de Météo-France à partir du cycle CY42_op1.*

Ce fichier est composé d'un enregistrement, possédant 4 champs séparés par un espace. Sa structure est la suivante :

- le suffixe de la base ECMA dans laquelle les données vont être stockées, en 8 caractères,
- l'extension du fichier binaire, en 8 caractères,
- le format du fichier binaire, en 8 caractères,
- le type de données ou de capteur, en 16 caractères.

*L'extension, le format du fichier binaire ainsi que le type de données/capteur doivent être les mêmes que ceux spécifiés dans le fichier **refdata**.*

Ci-dessous plusieurs exemples de fichiers **batormap** :

```
tovsa▫▫▫▫tovamsua▫BUFR▫▫▫▫▫amsua▫▫▫▫▫▫▫▫▫▫
conv▫▫▫▫▫europro▫▫BUFR▫▫▫▫▫europro▫▫▫▫▫▫▫▫▫
seviri▫▫▫sev000▫▫▫NETCDF▫▫▫seviri▫▫▫▫▫▫▫▫▫▫
```

4.3.4 Fichier ouloutput

Ce fichier ASCII a pour vocation de donner quelques informations sur le déroulement de ou des extractions demandées. Ci-dessous un exemple de fichier **ouloutput** généré dans le cadre opérationnel.

```
=== DEBUT DE PROGRAMME D'EXTRACTION ≡
               v. 1.2.1
===     POUR LE 20150908000000.     ≡

---> tentative d'extraction pour 'TEMPOMM' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:BUFR.temp
records read: 751
-- cutoff pour TEMPOMM : 20150909104701

---> tentative d'extraction pour 'SOLOMM' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 908
-- cutoff pour SOLOMM : 20150909104701
-- concatenation des SOLOMM activee dans BUFR.synop

---> tentative d'extraction pour 'SOLOMM' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 30837
-- cutoff pour SOLOMM : 20150909104703
```

```
-- concatenation des SOLOMM activee dans BUFR.synop

--- Nombre d'extractions demandees    :  3
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  3
--- Nombre de concatenations          :  1

--- Nombre de fichiers resultants     :  2

===  FIN DE PROGRAMME D'EXTRACTION  ===
```

4.4 Le cas particulier des NETCDF/HDF5

Un bloc d'extraction appliqué aux données au format NETCDF ou HDF5, contrairement à ceux concernant les données BUFR, peut générer plusieurs fichiers binaires. Pour ce faire, le nom de fichier indiqué par la clef `BINARY FILENAME` dans le FDEO est automatiquement suffixé par xxx (de 000 à 999). voir exemple Annexe – 3, page 36.

la longueur totale de l'extension d'un fichier binaire ne devant pas excéder 8 caractères, sa longueur doit être inférieure ou égale à 5 caractères dans la définition de la clef `BINARY FILENAME`.

5 le FDEO

5.1 Les types de phrases reconnues

Le FDEO comprend 4 types différents de phrases :

- le premier type de phrase est constitué par une *balise*, comme celles utilisées en HTML ou XML,
- le second type est composé par un couple *clef/valeur*, *clef/chaîne*, ou *clef/booléen*,
- le troisième par une *clef* suivie d'au moins 3 *chaînes*,
- enfin le dernier est un *commentaire*.

5.2 Règles d'écritures des phrases

Les 4 types de phrase sont régis par les règles d'écritures suivantes :

- une phrase ne peut comporter qu'une seule ligne,
- on ne peut avoir qu'une seule phrase par ligne,
- le caractère de séparation entre une *clef* et sa *valeur* (ou *chaîne* ou *booléen*) est le ':',
- une *chaîne* ne doit pas être encadrée par ' ou "
- une *clef* ou une *balise* peut être écrite en minuscules ou majuscules,
- une *chaîne* ou un *booléen* doit être écrit en respectant la casse prévue (voir « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr),
- une *balise*, une *clef*, une *chaîne*, un *booléen* ou une *valeur* peut être précédé ou suivi par des espaces (pas de tabulations !),
- un *commentaire* ne doit inclure ni nom de *balise* ni le caractère de séparation,
- on peut rajouter un *commentaire* à la suite d'une *balise* ou sur une ligne vierge, mais jamais après une phrase *clef/valeur*, *clef/chaîne(s)*, ou *clef/booléen*.

5.3 Listes des balises autorisées

5.3.1 <EXTRACTION> </EXTRACTION>

Ouvre et ferme un *bloc d'extraction*. Celui-ci peut comporter une ou plusieurs phrases de type *clef/valeur*, *clef/chaîne(s)*, *clef/booléen*, des *commentaires*, des *blocs colonne de données*, et des *blocs colonne d'en-tête.*

Syntaxe :

```
<EXTRACTION> [commentaire]
  [clef1 : valeur1]
  [clef1 : chaîne1]
  [clef1 : booléen1]
  ...
  [clefn : chaînen]
  [clefn : valeurn]
  [clefn : booléenn]
</EXTRACTION> [commentaire]
```

L'imbrication de « blocs d'extraction » est interdite.

5.3.2 <HEADING COLUMN> </HEADING COLUMN>

Ouvre et ferme un *bloc colonne d'en-tête* qui permet de récupérer une colonne d'en-tête BDMO (voir « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr) pour un type de données dans un fichier ASCII. Le format d'une colonne d'en-tête BDMO peut être un entier, un réel ou une chaîne de caractères.
Pour être valide, un *bloc colonne d'en-tête* doit renseigner les 2 *clés* `HEADING COL NAME` et `HEADING COL FMT`.
A noter que si l'on ne demande pas de fichier de sortie au format ASCII (cf. 5.4.29), les *blocs colonne d'en-tête* présents dans le **FDEO** seront ignorés.

Syntaxe :

```
<HEADING COLUMN> [commentaire]
  HEADING COL NAME : chaîne
  HEADING COL FMT  : chaîne
</HEADING COLUMN> [commentaire]
```

Un « bloc colonne d'en-tête » se trouve obligatoirement à l'intérieur d'un « bloc d'extraction ».

L'imbrication de « blocs colonne d'en-tête » est interdite.

5.3.3 <DATA COLUMN> </DATA COLUMN>

Ouvre et ferme un *bloc colonne de données* qui permet de récupérer une colonne ORACLE (voir « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr) pour un paramètre précis d'un type de données dans un fichier

ASCII. Toutes les colonnes ORACLE de données sont du type numérique.
Pour être valide, un *bloc colonne de données* doit renseigner les 3 *clés* `DATA COL NAME`, `DATA COL FMT`, et `DATA COL MISSING`.
A noter que si l'on ne demande pas de fichier de sortie au format ASCII (cf. 5.4.29), les *blocs colonne de données* présents dans le **FDEO** seront ignorés.

Syntaxe :

```
<DATA COLUMN> [commentaire]
  DATA COL NAME    : chaîne
  DATA COL FMT     : chaîne
  DATA COL MISSING : valeur
</DATA COLUMN> [commentaire]
```

Un « bloc colonne de données » se trouve obligatoirement à l'intérieur d'un « bloc d'extraction ».

L'imbrication de « blocs colonne de données » est interdite.

5.3.4 <BATORMAP> </BATORMAP>

Ouvre et ferme un *bloc batormap* qui permet la génération du fichier **batormap** (cf. 4.3.3) associé au fichier de données extrait.
Pour être valide, un *bloc batormap* doit renseigner les 2 *clés* `SENSOR` et `TARGET BASE`.
Dans le cadre d'une extraction ASCII seule, le fichier **batormap** ne sera pas généré, bien que ce bloc et ses 2 *clés* doivent être correctement renseignées.
Ces informations sont fondamentales, car réutilisées ensuite à l'entrée du programme **bator**.
En cas d'absence ou de malformation du *bloc batormap*, le *bloc d'extraction* courant sera ignoré.

Syntaxe :

```
<BATORMAP> [commentaire]
  SENSOR       : chaîne
  TARGET BASE  : chaîne
</BATORMAP> [commentaire]
```

Un bloc « batormap » se trouve obligatoirement à l'intérieur d'un « bloc d'extraction ».

L'imbrication de blocs « batormap » est interdite.

5.4 Liste des clés autorisées

La plupart des *clés* demandent à être renseignées avec des valeurs des booléens ou des chaînes qui dépendent directement de l'organisation de la BDMO ; la « Documentation Utilisateur BDMO », qui sert de référence, est disponible sur le serveur web de la DSI à l'adresse http://intradsi.meteo.fr.

5.4.1 OBS TYPE

Permet de définir le type BDMO que l'on désire extraire. Si cette *clef* est absente, la totalité du *bloc d'extraction* sera ignoré.

Syntaxe :

`[OBS TYPE : chaîne]`

5.4.2 UPPER LATITUDE

Permet de définir la latitude maximale (degrés décimaux, [-90°,+90°]) de la zone géographique pour laquelle on désire extraire des données. La valeur par défaut est +90°.
Si on a `UPPER LATITUDE` < `LOWER LATITUDE`, aucun fichier de données ne sera généré.

Syntaxe :

`[UPPER LATITUDE : valeur]`

Dans le cas d'une extraction à domaine limité de données satellites, pour éviter de perdre des observations, il est préférable de laisser la valeur de cette clef à +90° et d'utiliser la clef `PRECISE UPPER LATITUDE` *pour déterminer la limite nord du domaine.*

5.4.3 LOWER LATITUDE

Permet de définir la latitude minimale (degrés décimaux, [-90°,+90°]) de la zone géographique pour laquelle on désire extraire des données. La valeur par défaut est -90.
Si on a `UPPER LATITUDE` < `LOWER LATITUDE`, aucun fichier de données ne sera généré.

Syntaxe :

`[LOWER LATITUDE : valeur]`

Dans le cas d'une extraction à domaine limité de données satellites, pour éviter de perdre des observations, il est préférable de laisser la valeur de cette clef à -90° et d'utiliser la clef `PRECISE LOWER LATITUDE` *pour déterminer la limite sud du domaine.*

5.4.4 WESTER LONGITUDE

Permet de définir la longitude minimale (degrés décimaux, [-180°,+180°]) de la zone géographique pour laquelle on désire extraire des données. La valeur par défaut est -180°.
Si on a `WESTER LONGITUDE` > `EASTER LONGITUDE`, la zone définie passe par le méridien +/- 180.

Syntaxe :

`[WESTER LONGITUDE : valeur]`

Dans le cas d'une extraction à domaine limité de données satellites, pour éviter de perdre des observations, il est préférable de laisser la valeur de cette clef à -180° et d'utiliser la clef `PRECISE WESTER LONGITUDE` *pour déterminer la limite ouest du domaine.*

5.4.5 EASTER LONGITUDE

Permet de définir la longitude maximale (degrés décimaux, [-180°,+180°]) de la zone géographique pour laquelle on désire extraire des données. La valeur par défaut est +180°.
Si on a `WESTER LONGITUDE` > `EASTER LONGITUDE`, la zone définie passe par le méridien +/- 180.

Syntaxe :
`[EASTER LONGITUDE :` *valeur*`]`

Dans le cas d'une extraction à domaine limité de données satellites, pour éviter de perdre des observations, il est préférable de laisser la valeur de cette clef à +180° et d'utiliser la clef `PRECISE EASTER LONGITUDE` *pour déterminer la limite est du domaine.*

5.4.6 PRECISE UPPER LATITUDE

Permet, pour certains types de données BDMO, de définir la latitude maximale (degrés décimaux, [-90°,+90°]) de la zone géographique pour laquelle on désire faire une extraction. Aucune valeur par défaut n'est allouée à cette *clef*.
La valeur attribuée à cette *clef* est utilisée pour composer la clause WHERE de la requête SQL (*clef* `SQL WHERE`).
Si `PRECISE UPPER LATITUDE` < `PRECISE LOWER LATITUDE`, aucun domaine géographique ne sera inclus dans la clause WHERE.

Syntaxe :
`[PRECISE UPPER LATITUDE :` *valeur*`]`

5.4.7 PRECISE LOWER LATITUDE

Permet, pour certains types de données BDMO, de définir la latitude minimale (degrés décimaux, [-90°,+90°]) de la zone géographique pour laquelle on désire faire une extraction. Aucune valeur par défaut n'est allouée à cette *clef*.
La valeur attribuée à cette *clef* est utilisée pour composer la clause WHERE de la requête SQL (clef `SQL WHERE`).
Si `PRECISE UPPER LATITUDE` < `PRECISE LOWER LATITUDE`, aucun domaine géographique ne sera inclus dans la clause WHERE.

Syntaxe :
`[PRECISE LOWER LATITUDE :` *valeur*`]`

5.4.8 PRECISE WESTER LONGITUDE

Permet, pour certains types de données BDMO, de définir la longitude minimale (degrés décimaux, [-180°,+180°]) de la zone géographique pour laquelle on désire faire une extraction. Aucune valeur par défaut n'est allouée à cette *clef*.
La valeur attribuée à cette *clef* est utilisée pour composer la clause WHERE de la requête SQL (*clef* `SQL WHERE`).
Si on a `PRECISE WESTER LONGITUDE` > `PRECISE EASTER LONGITUDE`, la zone définie passe par le méridien +/- 180.

Syntaxe :

`[PRECISE WESTER LONGITUDE : valeur]`

5.4.9 PRECISE EASTER LONGITUDE

Permet, pour certains types de données BDMO, de définir la longitude maximale (degrés décimaux, [-180°,+180°]) de la zone géographique pour laquelle on désire faire une extraction. Aucune valeur par défaut n'est allouée à cette *clef*.
La valeur attribuée à cette *clef* est utilisée pour composer la clause WHERE de la requête SQL (clef `SQL WHERE`).
Si on a `PRECISE WESTER LONGITUDE` > `PRECISE EASTER LONGITUDE`, la zone définie passe par le méridien +/- 180.

Syntaxe :

`[PRECISE EASTER LONGITUDE : valeur]`

5.4.10 SQL WHERE

Comme son nom l'indique, cette *clef* permet de définir une requête where SQL sur les colonnes ORACLE de la table BDMO.
Par défaut, cette *clef* n'est pas définie. Dans ce cas elle sera ignorée.
En revanche, elle sera automatiquement renseignée, si l'une des *clés* `CUTOFF` ou `PRECISE OBS DATE` est renseignée, ou si un domaine géographique limité est défini par les *clés* `PRECISE UPPER LATITUDE`, `PRECISE LOWER LATITUDE`, `PRECISE WESTER LONGITUDE`, et `PRECISE EASTER LONGITUDE`.

Syntaxe :

`[SQL WHERE : chaîne]`

Si la longueur totale de chaîne dépasse 900 caractères, le bloc d'extraction courant sera ignoré.

Ne jamais définir le cutoff ou le domaine géographique « précis » d'extraction avec cette clef. Utilisez plutôt les clés prévues à cet effet.

5.4.11 PRECISE OBS DATE

Permet l'extraction d'un type de données pour une date précise. Celle-ci est exprimée relativement à la date de référence (réseau d'extraction) qui est la date pivot (cf. Annexe – 1, page 25). Sa valeur est utilisée pour composer la clause WHERE de la requête SQL (*clef* `SQL WHERE`).

NULL est la valeur prise par défaut.

Syntaxe :

`[PRECISE OBS DATE : chaîne]`

La date résultante donnée par la valeur de cette clef doit être comprise dans l'intervalle [DELTA BEFORE,DELTA AFTER].

5.4.12 DELTA BEFORE

Permet de définir la limite inférieure de la fenêtre temporelle d'extraction d'un type de données (date_de_l'observation >= date_pivot + limite_inférieure). Cette limite inférieure est exprimée relativement à la date de référence (réseau d'extraction) qui est la date pivot (cf. Annexe – 1, page 25).
heure –03 0000 est la valeur prise par défaut.

Syntaxe :

`[DELTA BEFORE : chaîne]`

5.4.13 BEFORE EXTENDED

Dans certains cas, des fichiers d'observations peuvent couvrir une longue période de temps, avec une heure d'observation 'optimale' en dehors de la fenêtre temporelle d'extraction définie par les clés DELTA BEFORE et DELTA AFTER.
BEFORE EXTENDED autorise la récupération de messages dont la date 'optimale' est en dehors de la fenêtre temporelle d'extraction mais qui possèdent des données appartenant à cette même fenêtre. Sa valeur doit être inférieure ou égale à celle définie par la clef DELTA BEFORE. Elle est exprimée relativement à la date de référence (réseau d'extraction) qui est la date pivot (cf. Annexe – 1, page 25).
La valeur par défaut est celle définie par la clef DELTA BEFORE.

Syntaxe :

`[BEFORE EXTENDED : chaîne]`

Actuellement (05/2017) seules les données MTVZA au format HDF5 sont concernées par cette clef (présence des colonnes jour_min, heure_min, jour_max, heure_max dans la table correspondante de la BDMO).

5.4.14 DELTA AFTER

Permet de définir la limite supérieure de la fenêtre temporelle d'extraction d'un type de données (date_de_l'observation <= date_pivot + limite_supérieure). Cette limite supérieure est exprimée relativement à la date de référence (réseau d'extraction) qui est la date pivot (cf. Annexe – 1, page 25).
heure +02 5959 est la valeur prise par défaut.

Syntaxe :

`[DELTA AFTER : chaîne]`

5.4.15 CUTOFF

Permet de définir l'heure de cutoff d'une extraction.
Si cette *clef* est renseignée, sa valeur est utilisée pour composer la clause WHERE de la requête SQL (*clef* `SQL WHERE`).
Si la *clef* n'est pas renseignée (ce qui est la valeur par défaut), ou si le format de la valeur n'est pas respecté, aucun cutoff ne sera inclus dans la clause WHERE.

Syntaxe :

`[CUTOFF : YYYYMMDDhhmmss]`

5.4.16 BINARY OUTPUT

Active ou désactive l'écriture du résultat de la demande d'extraction dans un fichier binaire (format BUFR ou NETCDF). Les états possibles sont Y ou N.
le défaut est N (désactivé, pas d'écriture dans fichier binaire).

Syntaxe :

`[BINARY OUTPUT : booléen]`

5.4.17 BINARY FILENAME

Définit le nom du fichier binaire qui doit être généré si `BINARY OUTPUT` est à Y. Si une concaténation est demandée (cf. 5.4.23), désigne le nom du fichier auquel on rajoute les données. La valeur attribuée par défaut est /dev/null.

Syntaxe :

`[BINARY FILENAME : chaîne]`

5.4.18 REFDATA

Cette *clef* permet la génération du fichier **refdata** associé au fichier d'extraction brut BUFR, NETCDF ou HDF5. Elle peut prendre 2 formes :
- une qui comporte 3 chaînes de caractères, la première correspondant au suffixe du fichier **refdata** qui sera créé, la seconde au format du fichier de données (BUFR/NETCDF/HDF5), et la troisième au capteur contenu dans le fichier.
- une autre syntaxe, plus récente, ne comporte que 2 chaînes de caractères, la première correspondant au format du fichier de données (BUFR/NETCDF/HDF5), et la seconde au capteur contenu dans le fichier.

Dans le cadre d'une extraction ASCII seule, le fichier **refdata** ne sera pas généré, bien que cette *clef* doive être correctement renseignée.
En cas d'absence de cette *clef*, le bloc d'extraction courant sera ignoré.

Syntaxe :

`[REFDATA : chaîne1 : chaîne2 : chaîne3]`
ou
`[REFDATA : chaîne1 : chaîne2]`

5.4.19 TARGET BASE

Cette *clef*, utilisée obligatoirement dans un *bloc batormap*, permet de spécifier le nom de la base ECMA qui va contenir les données du capteur défini par la *clef* `SENSOR`.

Syntaxe :

`[TARGET BASE : chaîne]`

5.4.20 SENSOR

Cette *clef*, obligatoirement dans un *bloc batormap*, permet de spécifier le capteur dont les données doivent être mises dans la base ECMA spécifiée par la *clef* `TARGET BASE`.

Syntaxe :

`[SENSOR : chaîne]`

`CHAÎNE` doit correspondre à l'un des capteurs définis dans la clef `REFDATA`.

5.4.21 MIN INDIC

Permet de définir le plus petit indicatif station souhaité du type BDMO que l'on désire extraire.
La chaîne utilisée par défaut est 0.

Syntaxe :

`[MIN INDIC : chaîne]`

5.4.22 MAX INDIC

Permet de définir le plus grand indicatif station souhaité du type BDMO que l'on désire extraire.
La chaîne utilisée par défaut est zzzzzzzzzz.

Syntaxe :

`[MAX INDIC : chaîne]`

5.4.23 CONCATENATION

Active ou désactive la possibilité de concaténer le fichier binaire résultat de l'extraction courante à un autre fichier binaire. Cette *clef* est utilisée pour obtenir un fichier « trié » d'un même type de données BDMO (optimisation du décodage BUFR). (cf. Annexe – 3 chapitres 2 et 4 pour avoir un exemple d'utilisation).
La valeur par défaut est N.
Cette option est forcée à N pour l'extraction des données radar (BUFR), et des données au format NETCDF ou HDF5.

Syntaxe :

`[CONCATENATION : booléen]`

5.4.24 SQL FROM

Comme son nom l'indique, cette *clef* permet de définir une requête from SQL.
Si cette *clef* n'est pas définie, elle est ignorée.

Syntaxe :

`[SQL FROM : chaîne]`

5.4.25 SQL ORDER BY

Comme son nom l'indique, cette *clef* permet d'obtenir une extraction ordonnée.
Si cette *clef* n'est pas définie, elle est ignorée.

Syntaxe :

`[SQL ORDER BY : chaîne]`

Un tri n'est effectif que si les données sont datées de la même journée.

5.4.26 RIGHTS

Permet de n'extraire que les données qui répondent à ce critère de « droits d'utilisation ».
La chaîne utilisée par défaut est OPER.

Syntaxe :

`[RIGHTS : chaîne]`

5.4.27 LAST ARRIVED

Permet de définir si on ne veut que les dernières versions valides des messages du type BDMO que l'on désire extraire. Les états possibles sont Y ou N.
En mode opérationnel, le défaut est Y.
En mode recherche, le défaut est N.

Syntaxe :

`[LAST ARRIVED : booléen]`

5.4.28 ARCHIVED DATA

Autorise ou non l'accès à la base archive. Les états possibles sont Y ou N.
En mode opérationnel, le défaut est N.
En mode recherche, le défaut est Y.

Syntaxe :

`[ARCHIVED DATA : booléen]`

5.4.29 ASCII OUTPUT

Active ou désactive l'écriture du résultat de la demande d'extraction dans un fichier ASCII. Les états possibles sont Y ou N. Si aucune colonne d'en-tête ni aucune colonne de données ne sont indiquées, le fichier généré sera vide.
En mode recherche et opérationnel, le défaut est N (désactivé).

Syntaxe :

`[ASCII OUTPUT : booléen]`

5.4.30 ASCII COMMENTED

Si `ASCII OUTPUT` est à Y, active ou désactive l'écriture du résultat de la demande d'extraction dans un fichier ASCII commenté, inutilisable par programme. Les états

possibles sont Y ou N.
En mode recherche et opérationnel, le défaut est N (désactivé).

Syntaxe :
`[ASCII COMMENTED : booléen]`

5.4.31 ASCII FILENAME

Définit le nom du fichier ASCII qui doit être généré si `ASCII OUTPUT` est à Y.
La valeur par défaut est /dev/null.

Syntaxe :
`[ASCII FILENAME : chaîne]`

5.4.32 SEPARATOR

Définit le caractère qui servira de séparateur de colonnes dans le fichier ASCII qui doit être généré si `ASCII OUTPUT` est à Y.
La caractère par défaut est la virgule.

Syntaxe :
`[SEPARATOR : caractère]`

5.4.33 PRINT MISSING VAL

Si `ASCII OUTPUT` est à Y, active ou désactive l'écriture des valeurs manquantes telles qu'elles sont définies dans les *blocs colonnes de données* par la *clef* `DATA COL MISSING`. Les états possibles sont Y ou N.
En mode recherche et opérationnel, le défaut est N (désactivé).

Syntaxe :
`[PRINT MISSING VAL : booléen]`

5.4.34 HEADING COL NAME

Cette *clef*, utilisée obligatoirement dans un *bloc colonne d'en-tête*, permet de spécifier le nom de la colonne d'entête BDMO à extraire dans le fichier ASCII (cf. « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr).

Syntaxe :
`[HEADING COL NAME : chaîne]`

5.4.35 HEADING COL FMT

Cette *clef*, utilisée obligatoirement dans un *bloc colonne d'en-tête*, permet de spécifier le format d'écriture de la colonne d'entête BDMO à extraire dans le fichier ASCII (cf. « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr).

Syntaxe :
`[HEADING COL FMT : chaîne]`

5.4.36 DATA COL NAME

Cette *clef*, utilisée obligatoirement dans un *bloc colonne de données*, permet de spécifier le nom de la colonne de données BDM à extraire dans le fichier ASCII (cf. « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr).

Syntaxe :

`[DATA COL NAME : chaîne]`

5.4.37 DATA COL FMT

Cette *clef*, utilisée obligatoirement dans un *bloc colonne de données*, permet de spécifier le format d'écriture de la colonne de données BDMO à extraire dans le fichier ASCII (cf. « Documentation Utilisateur BDMO » sur l'intranet DSI http://intradsi.meteo.fr).

Syntaxe :

`[DATA COL FMT : chaîne]`

5.4.38 DATA COL MISSING

Cette *clef*, utilisée obligatoirement dans un *bloc colonne de données*, permet de spécifier le format d'écriture des valeurs manquantes pour la colonne de données BDMO à extraire dans le fichier. La valeur précisée dans cette *clef* ne sera utilisée que si la *clef* `PRINT MISSING VAL` a la valeur Y.

Syntaxe :

`[DATA COL MISSING : chaîne]`

ANNEXES

ANNEXE – 1 : DATE PIVOT & METRONOME (EXTRAIT DU VADE-MECUM DSI)

1 Date-pivot et Profil de répétition

La date-pivot est caractéristique de l'exécution courante d'un travail, en ce sens qu'elle différencie cette exécution d'une autre exécution du même travail. On peut, dans un contexte Météo, pour un grand nombre de taches, rapprocher cette notion de celle de réseau.
Le premier principe à mémoriser est le suivant :
Les produits fabriqués par une exécution de date-pivot D seront datés de cette même date D.
C'est la raison pour laquelle le terme « date-pivot » est équivalent au terme « date des ressources produites ».
La date-pivot associée à un exécution d'un travail répétitif est extraite, dans la plupart des cas, d'un profil de répétition. Cette structure de données décrit la manière dont va se répéter un travail dans le temps.
La date-pivot va permettre, si on le désire, de calculer la date des données qu'on utilise, c'est-à-dire qu'on extrait d'une base de données. Ceci est possible en utilisant une expression d'utilisation de ressource, qui exprime l'écart temporel entre la date-pivot et la date de la ressource utilisée. Si on utilise plusieurs ensemble de données, il est possible, selon l'application, que l'expression soit la même pour tous, ou au contraire, qu'il faille une expression différente pour chacun.

2 Questions de base

Au début du portage d'une production sur SOPRANO, la première question à se poser est celle de la connaissance a priori de la date-pivot qu'on désire avoir dans son environnement d'exécution; en clair :

- quelle date (ou réseau) porte le produit que je fabrique ?

Cette question doit en amener une autre :

- avec quelle fréquence ce produit doit-il être fabriqué ?

Les réponses à ces deux questions vont permettre de spécifier un profil de répétition.
Ensuite, il s'agit de préciser les conditions de déclenchement du travail; la question peut se formuler ainsi :

- -quelles données me sont impérativement nécessaires ?

Il est tout à fait possible qu'il n'y ait pas de réponse claire, ou complète, à cette question ! La solution du déclenchement « sur heure », bien que peu souhaitable, doit alors être envisagée.
Mais, même dans ce dernier cas, on doit répondre à la question suivante :

- pour chacun des ensembles de données que j'utilise dans mon application, quelle est l'expression de date d'utilisation correspondante ?

Cette expression formalise simplement la date des données nécessaires par rapport à la date-pivot.
On voit donc qu'il est indispensable de bien intégrer la notion de date-pivot comme étant quelque chose de spécifique de votre application.

3 Le développement

Lorsque vous développez votre application sur les machines Alose ou Pagre, il faut avoir répondu aux questions précédentes; les essais que vous pourrez alors effectuer se feront alors en connaissance de cause. En effet, afin de reproduire le comportement opérationnel, il vous est demandé de créer vous-

même la date-pivot avant d'exécuter votre programme. Il faut savoir que, si vous ne le faites pas, les primitives ou commandes Metronome utiliseront la date système, ce qui n'est pas une approche idéale.
Par contre, dans un environnement de test, puis opérationnel, c'est le moniteur d'enchaînement de travaux qui générera lui-même la date-pivot.
Vous pouvez utiliser les services rendus par la bibliothèque Metronome soit au niveau code (C ou FORTRAN), soit au niveau shell-script UNIX. Toute la bibliothèque est documentée en ligne grâce à la commande `MAN`. Une introduction est disponible en faisant `MAN DMTINTRO`.

3.1 Un exemple

Soit une application dont les spécifications sont les suivantes :

- Je désire fabriquer un bulletin à partir de données de la BDAP.
- Ce bulletin doit être disponible chaque après-midi à partir des données du modèle Arpège de 12 UTC.

On en déduit :

- la caractéristique principale de mon travail sera d'être associé au réseau de 12 UTC; il sera donc daté 12 UTC chaque jour.
- j'ai besoin de certains champs issus de la BDAP dont l'expression d'utilisation de ressources est: "heure -0 0000" puisqu'il n'existe pas d'écart temporel entre ma date-pivot et les données que j'utilise.

Imaginons cette application sous forme de shell-script UNIX :

```
# mon_appli.sh
 ..................
# recuperation de la date des champs BDAP
EXPR="heure -0 0000"
DATE_BDAP=`dmtgetdutil "$EXPR"`
# extraction
.......... $DATE_BDAP.....
# execution
./mon_programme
# recuperation date-pivot pour dater mon produit
DATE_PROD=`dmtgetdprod`
# envoi BDPE
....... $DATE_PROD ....

Sur la machine de développement, je teste mon application en faisant :

$ export DMT_DATE_PIVOT=19941107120000
$ ./mon_appli.sh
```

4 Expression de la date pivot

Sa syntaxe est : `AAAAMMJJHHMMSS`.

5 Expression de date d'utilisation de ressource

Il s'agit d'une chaîne de caractères permettant d'exprimer l'écart temporel entre la date des ressources produites par un travail (date pivot) et la date des ressources utilisées par ce travail. Connaissant la date pivot et cette expression, on pourra facilement calculer la date des ressources utilisées (absolue), date qui devra être utilisée dans les requêtes aux bases de données.
l'expression est composée d'une partie « écart » (unité et valeur) et d'une partie « fixe » qui dépend de l'unité de l'écart. La syntaxe de cette expression est la suivante :

écart		partie fixe
unité	**valeur**	
an	[-]{[0-9]}+	MMJJhhmmss
mois		JJhhmmss
jour		hhmmss
heure		mmss
minute		ss
seconde		

Avec :

```
MM : 0[0-9] | 1[0-2]
JJ : 0[1-9] | [1-2](0-9] | 3[0-1] | dj
hh : [0-1][0-9] | 2[0-3]
mn : [0-5][0-9]
ss : [0-5][0-9]
```

Exemple 1 :
Un travail quotidien à pour date pivot 00h0000 et utilise les observations de 00h0000. L'expression d'utilisation peut être :

```
"heure -0 0000"
ou
"jour -0 0000"
```

Exemple 2 :
Un travail quotidien a pour date pivot 00h0000 et utilise les champs prévus du CEP de la veille à 12h0000. L'expression d'utilisation peut être :

```
"jour -1 120000"
ou
"heure -12 0000"
```

Exemple 3 :
Un travail mensuel s'exécute le premier jour de chaque mois et utilise des données du dernier jour du mois précédent, à 12h0000. L'expression d'utilisation sera :

```
"mois -1 dj120000"
```

ANNEXE – 2 : LES OUTILS DISPONIBLES

1 Fichier alim.awk & directives d'extraction

Le script awk est disponible sur supercalculateur pour les versions opérationnelle et double (quand une chaîne double existe !) en utilisant les commandes `GENV` et `GGET` mises à disposition. Il est inclus dans les packages de constantes du style extract.stuff.*modèle*.xx.gz.
A noter que les directives d'extractions sont également incluses dans ces packages.

2 Extractions via Olive

L'environnement Olive possède 3 trames de fond pour créer des expériences d'extraction de données ; une pour Arpège, une pour Aladin et une pour Arome. Pour plus de renseignements sur l'utilisation de ces configurations, contacter le support swapp.

3 Outils complémentaires

Il existe également un ensemble d'outils d'extractions complémentaires qui s'exécutent sur les machines SOPRANO et en dehors de l'environnement Olive. Ils sont disponibles sur Hendrix sous :
/home/guillaum/extractions/extractions.tar.gz
Une documentation d'installation et d'utilisation est disponible dans le package.

Ces outils autorisent l'extraction de données aux formats BUFR/NETCDF/HDF5/OBSOUL/GRIB :
– qu 'elles soient issues des BDMO opérationnelles (en ligne ou archive) ou intégration,
– pour un nombre quelconque de réseaux,
– sur une période quelconque,
– avec archivage (.tar ou non) optionnel des fichiers résultats sur Hendrix.

A noter que les directives d'extractions et namelists **Oulan** contenues dans ce package ne sont que des exemples. Les versions réellement utilisées sont à récupérer sur supercalculateur (cf. Annexe – 2, chapitre 1).

4 Outils de manipulations des fichiers BUFR

2 utilitaires de manipulation des fichiers BUFR, utilisables directement sur votre poste de travail, sont disponibles sur le site GMAPDOC (http://www.cnrm-game-meteo.fr/gmapdoc//spip.php?article227).

– DecodBufr est un décodeur de meta-fichier BUFR (bufr concaténés), comportant des options absentes de l'outil alternatif fourni par la DSI.

– SplitBufr découpe un méta-fichier BUFR (bufr concaténés) en fichiers BUFR muti-subsets ou mono-subset individuels.

Pour plus d'informations sur les options disponibles pour chacun de ces outils, se reporter au fichier README_FIRST présent dans chacun des packages.

Ces outils ne sont plus maintenus du fait que la librairie BUFR utilisée n'est également plus maintenue par le CEP. Des outils similaires (codes_count, codes_split_file, bufr_dump) sont disponibles dans la nouvelle API du CEP EcCodes.

ANNEXE – 3 : QUELQUES EXEMPLES

1 Extraction globale simple

On veut extraire les données TESAC sur tout le globe et sur la fenêtre temporelle par défaut [-3h, +3h[par rapport à la date pivot, qui vaut ici 20150908 à 0000 UTC. Les messages seront déposés dans le fichier **BUFR.tesac**. Le **FDEO** est alors de la forme :

```
<extraction>
 OBS TYPE        : TESAC
 BINARY OUTPUT   : Y
 BINARY FILENAME : BUFR.tesac
 REFDATA : BUFR : tesac
 batormap>
    SENSOR       : tesac
    TARGET BASE  : conv
 </batormap>
</extraction>
```

Les fichiers **ouloutput**, **batormap.tesac**, **refdata.tesac** résultants sont respectivement :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ≡
              v. 1.2.1
===     POUR LE 20150908000000.     ≡


---> tentative d'extraction pour 'TESAC' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:BUFR.tesac
records read: 654
-- cutoff pour TESAC : 20150909104711

--- Nombre d'extractions demandees    :  1
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  1

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  1

===  FIN DE PROGRAMME D'EXTRACTION  ≡
```

```
conv     tesac    BUFR     tesac
```

```
tesac    BUFR     tesac           20150908 00
```

2 Extraction globale avec concaténation et clause where

On veut extraire les données TOVSAMSUA issues de différents satellites selon un certain nombre de producteurs.
On souhaite obtenir ces données sur tout le globe et sur la fenêtre temporelle par défaut [-3h, +3h[par rapport à la date pivot, qui vaut ici 20150908 à 0000 UTC.
Pour ce faire il est nécessaire de réaliser une succession de requêtes, sachant que l'on veut obtenir un seul fichier de données en sortie, **BUFR.tovamsua**.
Le **FDEO** est alors de la forme :

```
<extraction>
 OBS TYPE         : TOVSAMSUA
 BINARY OUTPUT    : Y
 BINARY FILENAME  : BUFR.tovamsua
 CONCATENATION    : Y
 SQL WHERE        : and lltbufr_val in ('784')
 REFDATA          : BUFR : amsua
 <batormap>
    SENSOR        : amsua
    TARGET BASE   : tovsa
 </batormap>
</extraction>
<extraction>
 OBS TYPE         : TOVSAMSUA
 BINARY OUTPUT    : Y
 BINARY FILENAME  : BUFR.tovamsua
 CONCATENATION    : Y
 SQL WHERE        : and lltbufr_val in ('206','209','223') and prod_ori = 74
 REFDATA          : BUFR : amsua
 <batormap>
    SENSOR        : amsua
    TARGET BASE   : tovsa
 </batormap>
</extraction>
<extraction>
 OBS TYPE         : TOVSAMSUA
 BINARY OUTPUT    : Y
 BINARY FILENAME  : BUFR.tovamsua
 CONCATENATION    : Y
 SQL WHERE        : and lltbufr_val in ('206','209','223','4','3') and prod_ori = 211
 REFDATA          : BUFR : amsua
 <batormap>
    SENSOR        : amsua
    TARGET BASE   : tovsa
 </batormap>
</extraction>
<extraction>
 OBS TYPE         : TOVSAMSUA
 BINARY OUTPUT    : Y
 BINARY FILENAME  : BUFR.tovamsua
 CONCATENATION    : Y
 SQL WHERE        : and lltbufr_val in ('4','3') and prod_ori = 254 and prod_sec = 0
 REFDATA          : BUFR : amsua
 <batormap>
    SENSOR        : amsua
    TARGET BASE   : tovsa
 </batormap>
</extraction>
```

Les fichiers **ouloutput**, **batormap.tovamsua**, **refdata.tovamsua** résultants sont respectivement :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ≡
               v. 1.2.1
===     POUR LE 20150908000000.     ≡


---> tentative d'extraction pour 'TOVSAMSUA' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 295
-- cutoff pour TOVSAMSUA : 20150909105833
-- concatenation des TOVSAMSUA activee dans BUFR.tovamsua

---> tentative d'extraction pour 'TOVSAMSUA' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 1686
-- cutoff pour TOVSAMSUA : 20150909105857
-- concatenation des TOVSAMSUA activee dans BUFR.tovamsua
```

```
---> tentative d'extraction pour 'TOVSAMSUA' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 859
-- cutoff pour TOVSAMSUA : 20150909105901
-- concatenation des TOVSAMSUA activee dans BUFR.tovamsua

---> tentative d'extraction pour 'TOVSAMSUA' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 240
-- cutoff pour TOVSAMSUA : 20150909105907
-- concatenation des TOVSAMSUA activee dans BUFR.tovamsua

--- Nombre d'extractions demandees    :  4
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  4

--- Nombre de concatenations          :  1

--- Nombre de fichiers resultants     :  1

===  FIN DE PROGRAMME D'EXTRACTION  ≡
```

```
tovsa    tovamsua BUFR     amsua
```

```
tovamsua BUFR     amsua            20150908 00
```

3 Extractions avec heure précise et clauses where & order by

Dans l'exemple ci-dessous, on demande l'extraction des données de 3 sites RADAR :
- création d'un fichier par site,
- pour toutes les élévations valides et inférieure à 80°,
- en triant les élévations en ordre ascendant,
- toutes les données doivent être précisément datées à la date pivot (laquelle appartient bien à la fenêtre temporelle par défaut [-3h, +3h[. Le **FDEO** correspondant est :

```
<extraction>
 OBS TYPE           : RADAR
 BINARY FILENAME    : BUFR.07005
 BINARY OUTPUT      : Y
 MIN INDIC          : 07005
 MAX INDIC          : 07005
 PRECISE OBS DATE   : heure -00 0000
 SQL WHERE          : and elevation < 80. and flag_pretr<>1
 SQL ORDER BY       : order by elevation
 REFDATA            : BUFR : radarv
 <batormap>
    SENSOR          : radarv
    TARGET BASE     : radar1
 </batormap>
</extraction>
<extraction>
 OBS TYPE           : RADAR
 BINARY FILENAME    : BUFR.07027
 BINARY OUTPUT      : Y
 MIN INDIC          : 07027
 MAX INDIC          : 07027
 PRECISE OBS DATE   : heure -00 0000
```

```
 SQL WHERE          : and elevation < 80. and flag_pretr<>1
 SQL ORDER BY       : order by elevation
 REFDATA            : BUFR : radarv
 <batormap>
    SENSOR          : radarv
    TARGET BASE     : radar1
 </batormap>
</extraction>
<extraction>
 OBS TYPE           : RADAR
 BINARY FILENAME    : BUFR.07083
 BINARY OUTPUT      : Y
 MIN INDIC          : 07083
 MAX INDIC          : 07083
 PRECISE OBS DATE   : heure -00 0000
 SQL WHERE          : and elevation < 80. and flag_pretr<>1
 SQL ORDER BY       : order by elevation
 REFDATA            : BUFR : radarv
 <batormap>
    SENSOR          : radarv
    TARGET BASE     : radar1
 </batormap>
</extraction>
```

Les fichiers **ouloutput**, **batormap.07xxx**, **refdata.07xxx** résultants sont respectivement :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ≡
               v. 1.2.1
===     POUR LE 20151006000000.     ≡


---> tentative d'extraction pour 'RADAR' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:BUFR.07005
records read: 11
-- cutoff pour RADAR : 20151006102713
---> tentative de decompression du fichier radar.

---> tentative d'extraction pour 'RADAR' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:BUFR.07027
records read: 11
-- cutoff pour RADAR : 20151006102713
---> tentative de decompression du fichier radar.

---> tentative d'extraction pour 'RADAR' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:BUFR.07083
records read: 8
-- cutoff pour RADAR : 20151006102714
---> tentative de decompression du fichier radar.

--- Nombre d'extractions demandees    :  3
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  3

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  3

===  FIN DE PROGRAMME D'EXTRACTION  ≡
```

```
07005    BUFR     radarv          20151006 00
07027    BUFR     radarv          20151006 00
07083    BUFR     radarv          20151006 00
```

```
radar1   07005    BUFR     radarv
radar1   07027    BUFR     radarv
radar1   07083    BUFR     radarv
```

4 Extraction à domaine limité avec concaténation

On demande l'extraction sur un domaine limité des données GPSRO. Pour la définir, on utilise de préférence les clés du type « `PRECISE XXX YYY` » puisque la table contenant ces données possède les colonnes correspondantes. La fenêtre temporelle, est également redéfinie.
Pour des raisons d'optimisation, il est nécessaire de réaliser 2 requêtes successives, sachant que l'on veut obtenir un seul fichier de données en sortie, **BUFR.gpsro**. Le **FDEO** se présente alors comme suit :

```
<extraction>
 OBS TYPE                 : GPSRO
 BINARY OUTPUT            : Y
 CONCATENATION            : Y
 BINARY FILENAME          : BUFR.gpsro
 PRECISE UPPER LATITUDE   :  59.
 PRECISE LOWER LATITUDE   :  31.
 PRECISE WESTER LONGITUDE : -19.
 PRECISE EASTER LONGITUDE :  25.
 DELTA BEFORE             : heure -01 3000
 DELTA AFTER              : heure +00 2959
 SQL WHERE                : and lltbufr_val in ('723','4','3',42,'43','820','786')
 REFDATA                  : BUFR : gpsro
 <batormap>
    SENSOR                : gpsro
    TARGET BASE           : gps
 </batormap>
</extraction>
<extraction>
 OBS TYPE                 : GPSRO
 BINARY OUTPUT            : Y
 CONCATENATION            : Y
 BINARY FILENAME          : BUFR.gpsro
 PRECISE UPPER LATITUDE   :  59.
 PRECISE LOWER LATITUDE   :  31.
 PRECISE WESTER LONGITUDE : -19.
 PRECISE EASTER LONGITUDE :  25.
 DELTA BEFORE             : heure -01 3000
 DELTA AFTER              : heure +00 2959
 SQL WHERE                : and lltbufr_val in ('740','741','743','744','745')
 REFDATA                  : BUFR : gpsro
 <batormap>
    SENSOR                : gpsro
    TARGET BASE           : gps
 </batormap>
</extraction>
```

Seul le fichier **ouloutput** est généré car il n'y a pas de données répondant aux critères de la requête :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ===
               v. 1.2.1
===     POUR LE 20151006000000.     ===


---> tentative d'extraction pour 'GPSRO' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 0
```

```
*** WARNING : fichier vide.

---> tentative d'extraction pour 'GPSRO' sur /usr/local/sopra/neons_db_bdm
nom fich:/dev/null
nom fich bufr:tmp_Bufr_99999
records read: 0
*** WARNING : fichier vide.

--- Nombre d'extractions demandees    :  2
----------   extractions erronees     :  0
----------   extractions fichier vide :  2
----------   extractions sans fichier :  0

----------   extractions avec fichier :  0

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  0

===  FIN DE PROGRAMME D'EXTRACTION  ≡
```

5 Extraction de données au format NETCDF (exemple valable pour HDF5)

Dans cet exemple, on désire obtenir les données SEVIRI au format NETCDF.
- On souhaite obtenir ces données sur tout le globe et sur la fenêtre temporelle [-0h30, +0h30[par rapport à la date pivot, qui vaut ici 20151018 à 0000 UTC.
- On suppose que ce type de données n'est disponible que dans la BDMO d'intégration (cf. 4.1).
Ici, sont générés 2 fichiers de données, **NETCDF.sev000** et **NETCDF.sev001**

```
<extraction>
   OBS TYPE         : SEVIRI
   BINARY OUTPUT    : Y
   BINARY FILENAME  : NETCDF.sev
   DELTA BEFORE     : heure -01 3000
   DELTA AFTER      : heure +00 2959
   REFDATA : NETCDF : seviri
   <batormap>
      SENSOR        : seviri
      TARGET BASE   : seviri
   </batormap>
</extraction>
```

Les fichiers **ouloutput**, **refdata.sevxxx**, **batormap.sevxxx**, résultants sont respectivement :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ===
               v. 1.2.1
===     POUR LE 20151018000000.     ===


---> tentative d'extraction pour 'SEVIRI' sur /usr/local/sopra/neons_db_bdm.intgr
nom fich:/dev/null
nom fich bufr:NETCDF.sev
records read: 2
-- cutoff pour SEVIRI : 20151019082741

--- Nombre d'extractions demandees    :  1
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  1

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  1

===  FIN DE PROGRAMME D'EXTRACTION  ===
```

```
sev000   NETCDF   seviri          20151018 00
sev001   NETCDF   seviri          20151018 00
```

```
seviri   sev000   NETCDF   seviri
seviri   sev001   NETCDF   seviri
```

6 Extraction de données MTVZA au format HDF5.

Dans cet exemple on veut obtenir tous les fichiers qui possèdent au moins 1 observation dans le fenêtre temporelle d'extraction par défaut [-3h, +3h[.
La date d'observation 'optimale' de ces fichiers correspond toujours au premier point de mesure. De plus leur contenu peut couvrir une période de temps supérieure à 2 heures. C'est pourquoi, on va utiliser la clef `BEFORE EXTENDED` pour être certain de récupérer tous les fichiers pertinents.
- On suppose que ce type de données n'est disponible que dans la BDMO d'intégration (cf. 4.1).
Ici, sont générés 8 fichiers de données, **HDF5.mtvza000** à **HDF5.mtvza007**

```
<extraction>
   OBS TYPE        : MTVZA
   BINARY OUTPUT   : Y
   BINARY FILENAME : HDF5.mtvza
   REFDATA         : HDF5 : mtvza
   BEFORE EXTENDED : heure -06 0000
  <batormap>
      SENSOR       : mtvza
      TARGET BASE  : mtvza
   </batormap>
</extraction>
```

Les fichiers **ouloutput**, **refdata.mtvzaxxx**, **batormap.mtvzaxxx**, résultants sont respectivement :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ===
               v. 1.2.4
===     POUR LE 20170531000000.     ===


---> tentative d'extraction pour 'MTVZA' sur /usr/local/sopra/neons_db_bdm.intgr
nom fich:/dev/null
nom fich bufr:HDF5.mtvza
records read: 8
-- cutoff pour MTVZA : 20170531133230

--- Nombre d'extractions demandees    :  1
----------   extractions erronees     :  0
----------   extractions fichier vide :  0
----------   extractions sans fichier :  0

----------   extractions avec fichier :  1

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  1

===  FIN DE PROGRAMME D'EXTRACTION  ===
```

```
mtvza000 HDF5     mtvza           20170531 00
mtvza001 HDF5     mtvza           20170531 00
mtvza002 HDF5     mtvza           20170531 00
mtvza003 HDF5     mtvza           20170531 00
mtvza004 HDF5     mtvza           20170531 00
mtvza005 HDF5     mtvza           20170531 00
mtvza006 HDF5     mtvza           20170531 00
```

```
mtvza007 HDF5     mtvza            20170531 00
```

```
mtvza    mtvza000 HDF5     mtvza
mtvza    mtvza001 HDF5     mtvza
mtvza    mtvza002 HDF5     mtvza
mtvza    mtvza003 HDF5     mtvza
mtvza    mtvza004 HDF5     mtvza
mtvza    mtvza005 HDF5     mtvza
mtvza    mtvza006 HDF5     mtvza
mtvza    mtvza007 HDF5     mtvza
```

7 Demande d'extraction en ASCII – mode recherche

Dans cet exemple, on désire obtenir uniquement un fichier ASCII, exploitable informatiquement, et qui contient certains paramètres contenues dans les messages SYNOP sur tout le globe. Comme il s'agit de la version « recherche », on peut spécifier le cutoff explicitement. De plus, aucune sortie binaire n'est demandée, les fichiers **refdata** et **batormap** ne sont donc pas générés.

```
# Fichier de directives pour le script d'extraction VERSION TEST POUR ASCII

<extraction>
   OBS TYPE        : SYNOPA
   ARCHIVED DATA   : Y
   LAST ARRIVED    : N
   CUTOFF          : 20080410000000
   BINARY OUTPUT   : N
   ASCII OUTPUT    : Y
   ASCII FILENAME  : ASCII.SYNOP
   PRINT MISSING VAL : Y
   REFDATA : ASCII : null
   <batormap>
      SENSOR               : synop
      TARGET BASE          : synop
   </batormap>
 <HEADING COLUMN>
   HEADING COL NAME : lltbufr_val
   HEADING COL FMT  : %-10s
 </HEADING COLUMN>
 <HEADING COLUMN>
   HEADING COL NAME : date
   HEADING COL FMT  : %-14s
 </HEADING COLUMN>
 <DATA COLUMN>
   DATA COL NAME    : dd
   DATA COL FMT     : %3.0f
   DATA COL MISSING : -999.
 </DATA COLUMN>
 <DATA COLUMN>
   DATA COL NAME    : ff
   DATA COL FMT     : %4.1f
   DATA COL MISSING : -999.9
 </DATA COLUMN>
 <DATA COLUMN>
   DATA COL NAME    : t
   DATA COL FMT     : %5.2f
   DATA COL MISSING : -999.99
 </DATA COLUMN>
</extraction>
```

Dans le cas d'une demande d'extraction uniquement ASCII, le compteur d'extraction avec fichier n'est pas incrémenté. Le fichier **ouloutput** résultant ressemble donc à :

```
=== DEBUT DE PROGRAMME D'EXTRACTION ===
              v. 1.2.1
===     POUR LE 20151006000000.     ===
```

```
---> tentative d'extraction pour 'SYNOPA' sur /usr/local/sopra/neons_db_bdm
nom fich:ASCII.SYNOP
nom fich bufr:/dev/null
records read: 17825
*** WARNING : fichier vide.

--- Nombre d'extractions demandees    :  1
----------   extractions erronees     :  0
----------   extractions fichier vide :  1
----------   extractions sans fichier :  0

----------   extractions avec fichier :  0

--- Nombre de concatenations          :  0

--- Nombre de fichiers resultants     :  0

===  FIN DE PROGRAMME D'EXTRACTION  ===
```

Ci-dessous un extrait du fichier de données **ASCII.SYNOP** obtenu :

```
02944     ,20151005210000, 20, 3,0,276,65,
02947     ,20151005210000,340, 2,0,272,85,
02958     ,20151005210000,360, 3,0,276,65,
02963     ,20151005210000, 10, 5,0,273,55,
02965     ,20151005210000,330, 1,0,275,05,
02967     ,20151005210000, 20, 8,0,279,15,
02974     ,20151005210000,360, 4,0,274,95,
02976     ,20151005210000,360, 5,0,278,05,
02980     ,20151005210000, 60, 7,0,281,45,
02982     ,20151005210000, 30, 8,0,279,35,
02984     ,20151005210000, 40, 7,0,279,25,
02987     ,20151005210000, 40,10,0,280,05,
02991     ,20151005210000,360, 1,0,276,85,
02992     ,20151005210000, 10, 8,0,279,35,
02993     ,20151005210000, 70, 9,0,282,55,
02720     ,20151005210000,290, 3,0,271,55,
02721     ,20151005210000, 40, 9,0,280,35,
02737     ,20151005210000,360, 2,0,271,35,
02748     ,20151005210000,360, 4,0,275,85,
02751     ,20151005210000, 60, 5,0,277,25,
02757     ,20151005210000, 30, 5,0,279,15,
02759     ,20151005210000,360, 2,0,273,15,
02777     ,20151005210000,350, 3,0,273,45,
02795     ,20151005210000, 20, 6,0,278,75,
02816     ,20151005210000,300, 1,0,269,65,
02824     ,20151005210000,320, 2,0,270,05,
02834     ,20151005210000, 10, 3,0,272,75,
```