Hierarchical Data Format

1. **Généralités**
2. **Structure des fichiers HDF**
3. **L'API HDF**
4. **HDF-EOS**
5. **Outils associés**

1. Généralités

Formats de données standardisés (1)

- Volumes de données de plus en plus importants dans tous les domaines : industrie, recherche, …
- Définition, acquisition, traitement, archivage et distribution de ces informations dans des environnements de plus en plus complexes (hétérogènes, distribués, …)
- Nécessité de consolider le lien entre la perception du concepteur et la compréhension de l'utilisateur, fragile dans le cas des formats « maison »
  - absence, insuffisance, indisponibilité de la documentation
  - hétérogénéité des documents
  - apprentissage incontournable
  - développement de code spécifique souvent nécessaire

Formats de données standardisés (2)

Approche cohérente : un référentiel unique commun à tous les acteurs du cycle de vie de la donnée

Les nombreux formats de données standardisés (***FDS***) proposent une réponse (partielle) à cette problématique :

- **CDF** - ***C****ommon* ***D****ata* ***F****ormat* ; physique de l'espace
- **NetCDF** - ***N****etwork* ***C****ommon* ***D****ata* ***F****ormat* ; climatologie, météorologie, océanographie
- **FITS** - ***F****lexible* ***I****mage* ***T****ransport* ***S****ystem* ; astronomie, astrophysique, physique solaire
- **HDF** - ***H****ierarchical* ***D****ata* ***F****ormat* ; observations spatiales, climatologie, océanographie, environnement

C’est quoi, HDF ?

Format de fichier standard et public créé initialement en 1987 par le ***N.C.S.A.*** (**N**ational **C**enter for **S**upercomputing **A**pplications) et maintenu par le ***HDF Group*** (2006) pour :

- Support de la majorité des types de données et méta-données utilisées par la communauté scientifique
- Stockage efficace et accès à de très grands volumes de données
- Indépendance vis-à-vis des plates-formes et systèmes
- Evolutivité, compatibilité avec d’autres formats standards
- Auto-documentation

Intérêts de HDF (1)

- Souplesse :
  HDF accepte de nombreux modèles de données différents qu'il peut stocker dans un même fichier (ou plusieurs depuis la version HDF3.3 de 1993)
- Flexibilité :
  Stockage dans un même fichier (ou plusieurs) de données reliées conceptuellement mais de sources et formats physiques différents
- Abstraction :
  La donnée est connue par sa représentation conceptuelle et manipulée grâce à une interface logicielle (séparation contenu / contenant)

Intérêts de HDF (2)

- Portabilité :
  Indépendance du format HDF vis-à-vis des plates-formes, des systèmes d'exploitation et des langages
- Hiérarchisation des données :
  Possibilité de regrouper en « *Vgroups* » des ensembles de données logiquement reliées, eux-mêmes pouvant contenir des Vgroups.
- Auto-description des données :
  Une application peut connaître la structure et utiliser le contenu d'un fichier HDF sans information extérieure (≠ formats « propriétaires »).

Intérêts de HDF (3)

- Auto-documentation :
  L'utilisateur peut attacher des « attributs » et/ou des « annotations » (≈ méta-données) aux diverses unités de données du fichier ou au fichier lui-même
- Extensibilité :
  Adaptation à la manipulation de n'importe quel modèle de données par la création de nouveaux types de données et de nouvelles interfaces (cf. HDF-EOS)
- Standardisation des formats et descriptions de beaucoup de types de données d'usage scientifique courant
- Domaine public

Différentes versions de HDF (1)

HDF = HDF4 :

1996 pour la 4.0, 02/2010 pour la 4.2r5, version actuelle considérée comme « finalisée »

Limitations :

- maximum de 20 000 objets complexes par fichier
- taille maximale d'un fichier HDF : 2 Go (problème réglé en partie par les interfaces multi-fichiers)
- modèle de données inconsistant
- la bibliothèque est ancienne, complexe à cause des modifications et enrichissements successifs, ne supporte pas les E/S parallèles, et mal les applications « threadées »
- tableaux de types composites limités à une dimension
- modifications/suppressions difficiles ou impossibles

Différentes versions de HDF (2)

Pour résoudre ces problèmes → HDF5 :

- 1998, 11/2009 pour la 1.6.10, version actuelle
- pas de compatibilité HDF4→HDF5 (coexistence possible)
- plus de limitation de nombre d'objets et de taille de fichier (sauf liée au système)
- modèle de données plus simple et consistant : 2 types de base uniquement au lieu de 6 dans HDF4 (*voir plus loin*) :
  - tableau multi-dimensionnel d'enregistrements
  - structure de regroupement
- une bibliothèque plus simple et mieux conçue, supportant les E/S parallèles et les « threads ».
- …

HDF4 continue à être maintenue (mais plus enrichie)

Différentes versions de HDF (3)

HDF-EOS :

- (1996, 07/2000 pour la 2.7 actuelle)
- développé dans le cadre du projet ***EOSDIS*** (***E****arth* ***O****bserving* ***S****ystem* ***D****ata and* ***I****nformation* ***S****ystem*)
- extension de HDF par ajout de types de données de base et d'une API adaptée (*voir plus loin*)
- HDF5-EOS : même extension pour HDF5

Conclusion :

$$\begin{array}{ccc} \text{HDF} & \subset & \text{HDF-EOS} \\ \neq & & \neq \\ \text{HDF5} & \subset & \text{HDF5-EOS} \end{array}$$

Cette présentation concerne uniquement HDF(4)

Disponibilité

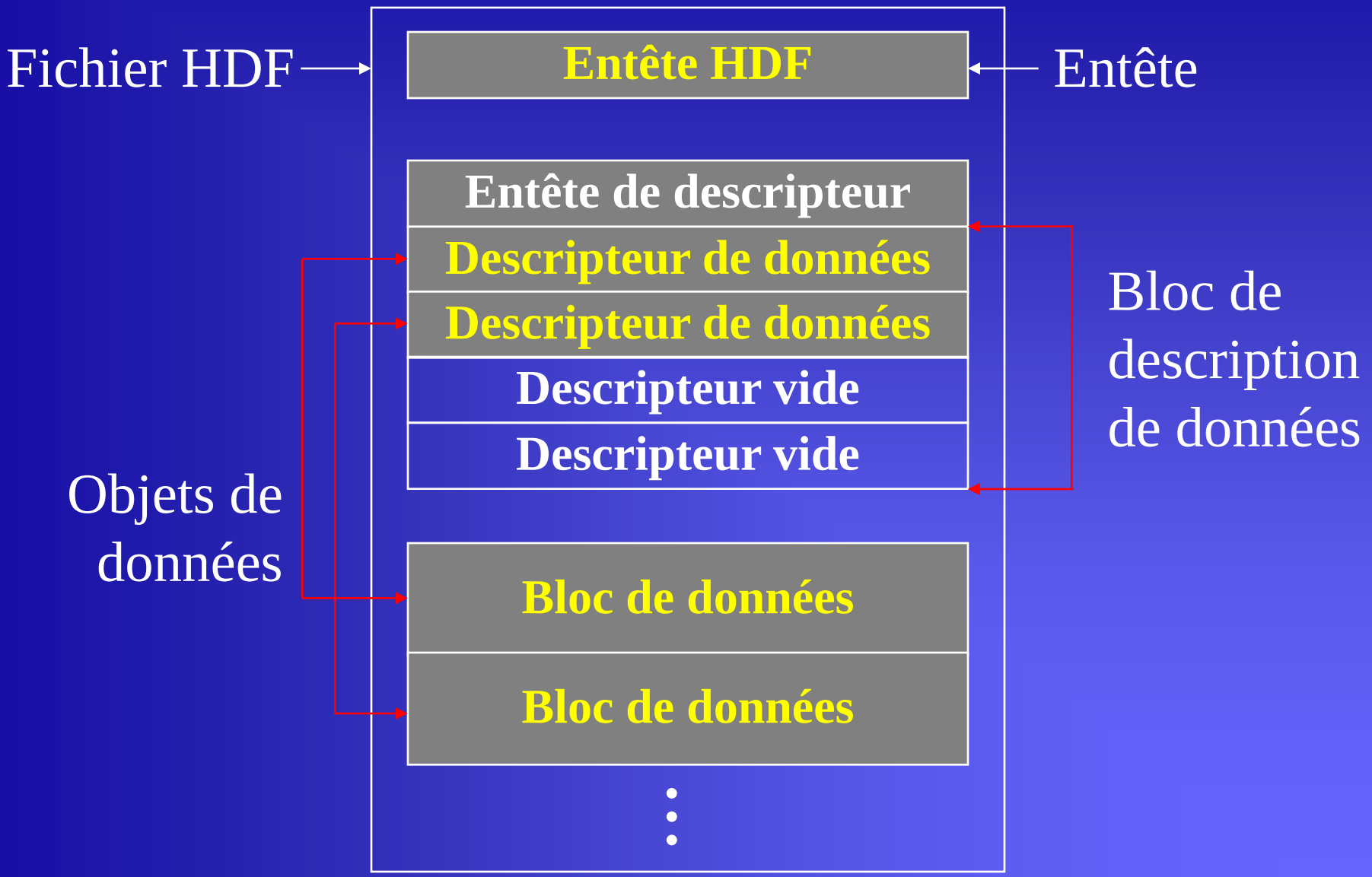
Très nombreuses plates-formes 32 ou 64 bits supportées :

- Windows 98/NT/XP/Vista
- Mac OS X
- Linux
- Sun OS
- AIX
- …

Voir : http://www.hdfgroup.org/release4/platforms.html

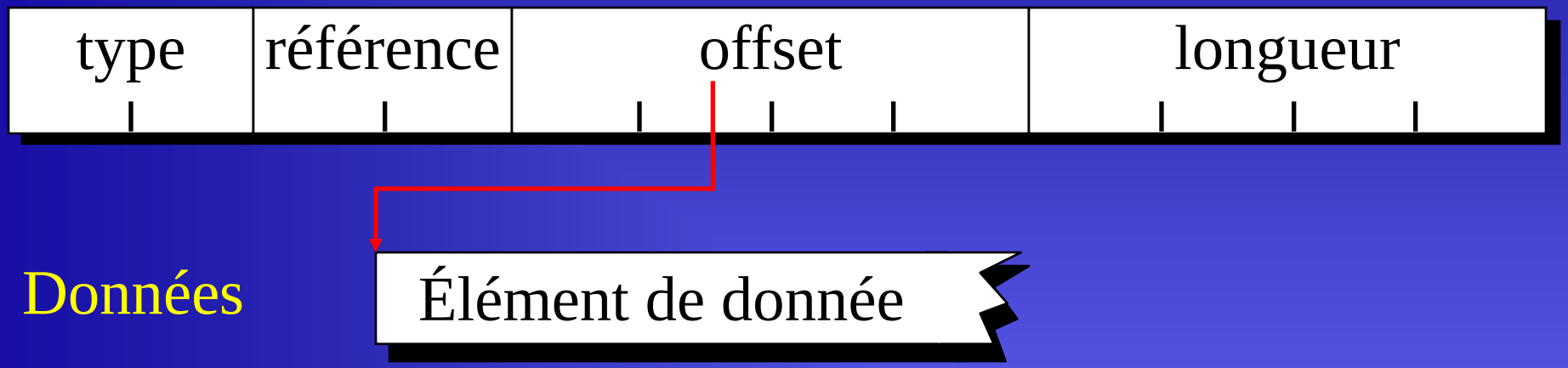
2. Structure des fichiers HDF

Structure d'un fichier HDF



Structure de l'objet de données HDF

Descripteur de données : 12 octets



- type : type de donnée de base (il y en a 6)
- référence : identifie uniquement l'élément de donnée
- offset : de la donnée par rapport au début de fichier
- longueur : de la donnée, en octets

Les 6 structures de base HDF

palette

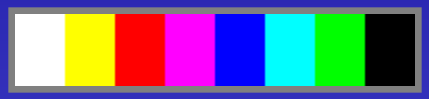
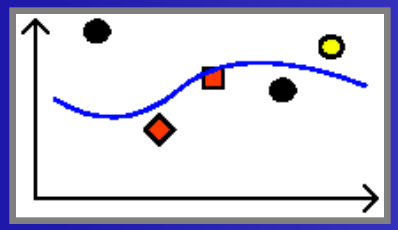
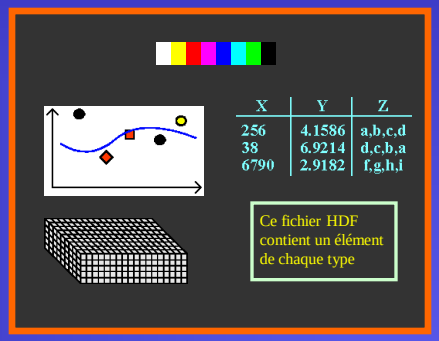


image raster

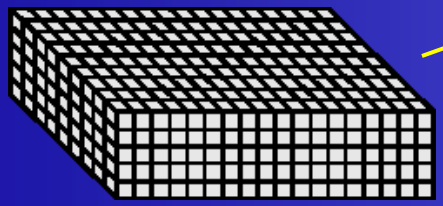


Vdata : table

X	Y	Z
256	4.1586	a,b,c,d
38	6.9214	d,c,b,a
6790	2.9182	f,g,h,i



Vgroup : groupe de structures HDF

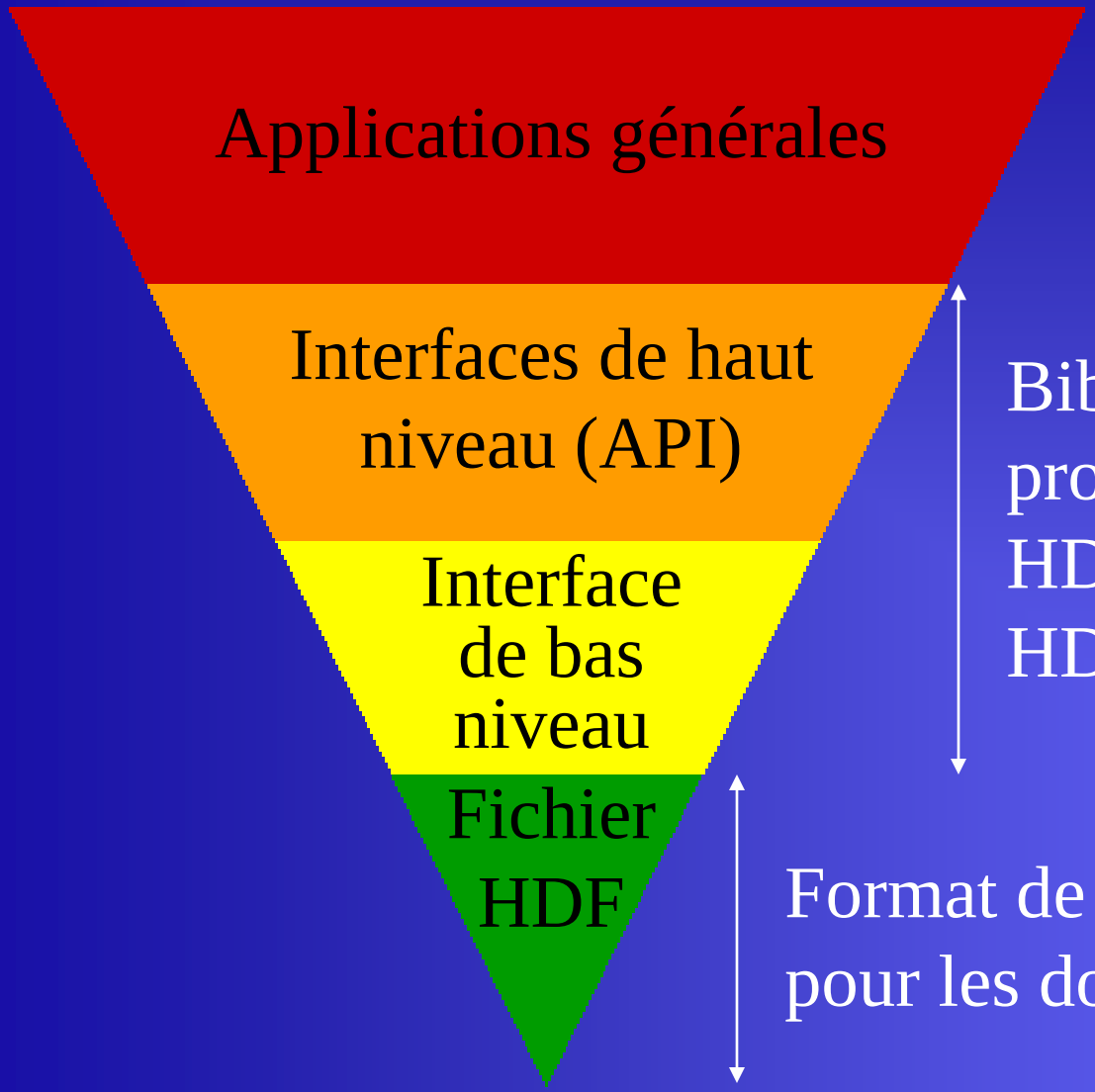


SDS (Scientific Data Set) : tableau multi-dimensionnel

Ce fichier HDF contient un élément de chaque type

annotation

Les différentes couches de HDF (1)

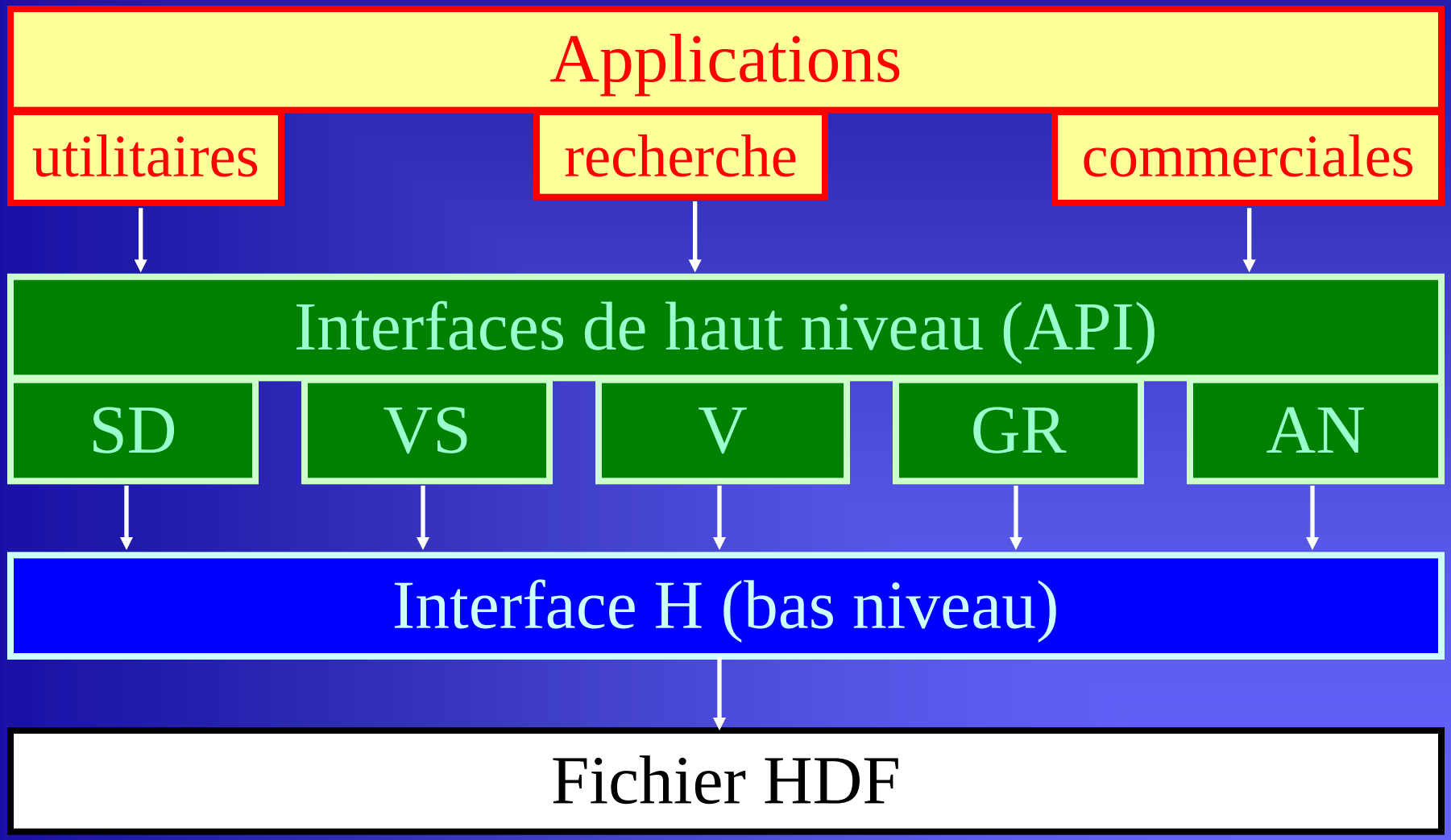


Manipulation, visualisation des données

Bibliothèques de programmes (HDF4, HDF5, HDF4-EOS, HDF5-EOS, ...)

Format de fichier physique pour les données scientifiques

Les différentes couches de HDF (2)



3. L'API HDF

Types de données HDF (1)

- char8 : caractère sur 8 bits
- uchar8 : caractère sur 8 bits non signé
- intn : entier « natif »
- int8 : entier sur 8 bits
- uint8 : entier sur 8 bits non signé
- int16 : entier sur 16 bits
- uint16 : entier sur 16 bits non signé
- int32 : entier sur 32 bits
- uint32 : entier sur 32 bits non signé
- float32 : flottant sur 32 bits
- float64 : flottant sur 64 bits

Types de données HDF (2)

- Type intn : code de retour en général
- Toutes les conversions nécessaires sont transparentes
  ex : big-endian (défaut) ↔ little-endian

Interface H - bas niveau (1)

- Hopen : fournit un moyen d'accès à un fichier HDF en chargeant en mémoire tous les blocs de description de données
- Hclose : ferme le fichier HDF
- Hgetlibversion : retourne des informations sur la version de la bibliothèque HDF
- Hgetfileversion : retourne des informations sur la version d'un fichier HDF
- Hishdf : indique si un fichier est au format HDF
- HEstring : rend le message associé à un code d'erreur
- …

Interface H - bas niveau (2)

Exemple en C :

```
int32 Hopen(char *filename, intn access, int16 ndds);
```

- *filename* (E) : chemin du fichier à ouvrir
- *access* (E) : code du type d'accès (**DFACC_CREATE**, **DFACC_READ**, **DFACC_WRITE**)
- *ndds* (E) : nombre de descripteurs par bloc (création)

Retourne un identificateur de fichier ou **FAIL** (-1) si échec

Même exemple en Fortran :

```
integer function hopen(filename, access, ndds)
character*(*) filename
integer access, ndds
```

Les interfaces de haut niveau (1)

- Interface AN :
  pour lire et écrire des annotations
- Interface SD :
  pour stocker, gérer et extraire des tableaux multi-dimensionnels de caractères ou de valeurs numériques (i.e. les ***SDS*** ou ***Scientific Data Sets***)
- Interfaces VS, VF, VSQ:
  VS pour stocker, gérer et extraire des données de types différents agencées sous la forme de tableaux de structures (i.e. les ***Vdata***), VF pour manipuler les champs d'un Vdata, VSQ pour interroger un Vdata

Les interfaces de haut niveau (2)

- Interface GR :
  pour stocker, gérer et extraire des images raster ainsi que les palettes de couleurs associées
- Interface V :
  pour créer et gérer des groupes de structures HDF (i.e. les ***Vgroups***)

Toutes sont des interfaces multi-fichiers.

D'autres interfaces mono-fichier existent ; elles ne sont là que pour des raisons de compatibilité ascendante : DFAN, DFSD, DFR8, DF24, DFP.

Les attributs

Possibilité d'associer des ***attributs*** (≈ méta-données) à un SDS, une dimension, un Vdata, une image raster, un Vgroup ou au fichier HDF lui-même :

- attributs prédéfinis :
  - "Label", "Unit", "Format" (SDS ou dimension)
  - "Fill Value" (SDS ou image raster)
  - "Coordinate System", "Range", "Calibration" (SDS)
- attributs « utilisateur » (tous)

Autres fonctionnalités

- Interface de gestion des erreurs (HE)
- Possibilité de ***nommer*** les dimensions des SDS, de les ***partager*** entre différents SDS, de leur associer un vecteur de « coordonnées »
- Possibilité pour un SDS d'avoir une dimension illimitée (extension possible sur cette dimension)
- Possibilité de ***compresser*** (avec choix de l'algorithme) et/ou de ***morceler*** (stockage en blocs de taille fixe ou ***chunks***) un SDS ou une image raster

Exemple : lecture en C d'un SDS (1)

```
char* fmod = "MOD03.A2003101.1225.004.2003101213042.hdf";
int32 sd_id, s_idx, s_id, ndim, typ, na, dim[2];
float32 *data;

/* On initialise l'interface SD sur le fichier HDF */
check(sd_id = SDstart(fmod, DFACC_READ) >= 0);

/* On récupère la référence du SDS par son nom */
check(s_idx = SDnametoindex(sd_id, "Latitude") >= 0);

/* On récupère un accès sur le SDS */
check(s_id = SDselect(sd_id, s_idx) >= 0);

/* On récupère les dimensions du SDS */
check(SDgetinfo(s_id, NULL, &ndim, dim, &typ, &na) >= 0);

/* On vérifie la validité du nombre de dimensions */
check(ndim == 2);
```

.../...

Exemple : lecture en C d'un SDS (2)

```
/* On alloue l'espace pour les données */
check(data = (VOIDP)calloc(dim[0]*dim[1],
      sizeof(float32)) != NULL);

/* Lecture des données */
start[0] = start[1] = 0;
check(SDreaddata(s_id, start, NULL, dim, data) >= 0);

/* On termine l'accès au SDS */
check(SDendaccess(s_id) >= 0);

/* On termine l'interface SD */
check(SDend(sd_id ) >= 0);
```

Remarque : le fichier n'a pas été ouvert par un **`Hopen`**. Ici, c'est fait implicitement par **`SDstart`** (et le **`Hclose`** par **`Sdend`**). Ce n'est vrai que pour l'interface SD...

Même exemple en Python

```
fmod = "MOD03.A2003101.1225.004.2003101213042.hdf

try:

   # On initialise l'interface SD sur le fichier HDF
   fhdf = SD(fmod, SDC.READ)

   # On récupère un accès sur le SDS
   sds = fhdf.select("Latitude")

   # Lecture des données
   data = sds.get()

   # On termine l'accès au SDS
   sds.endaccess()

   # On termine l'interface SD
   fhdf.end()

except:

   # Message d'erreur
   print "Erreur lecture %s" % fmod
```

4. HDF-EOS

Pourquoi HDF-EOS ?

Types de données de base HDF mal adaptés aux données EOS, surtout dans le cas de données géo-localisées et/ou ordonnées chronologiquement :

- donnée ponctuelle
- grille de projection
- scan ou profil de satellite défilant
- …

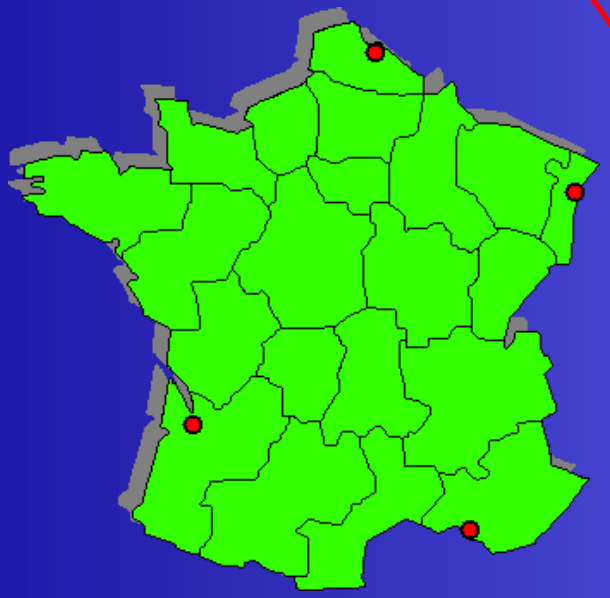
→ 3 nouveaux types de données et interfaces adaptées :

- point (PT)
- swath (SW)
- grid (GD)

L’interface PT (point)

Support des données géo-localisées mais non organisées spatio-temporellement.

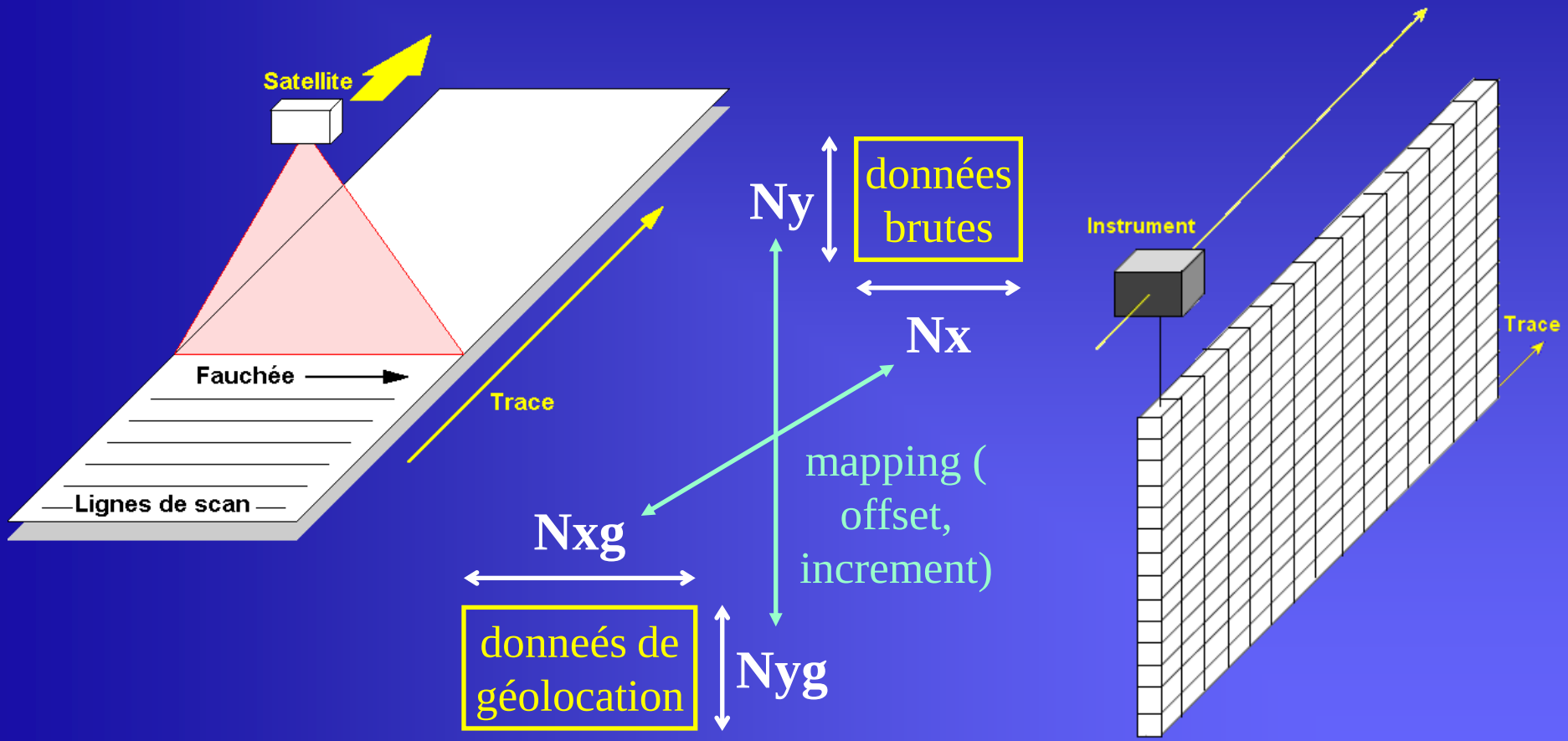
Station	Lat	Lon
Lille	50.39	3.05
Strasbourg	48.35	4.45
Bordeaux	44.50	-0.74
Marseille	43.18	5.22



Heure	Temp
08:00	4
09:00	6
10:00	7
08:00	-2
09:00	0
10:00	1
11:00	3
10:00	12
11:00	13
12:00	15
13:00	15
14:00	16
06:00	12
07:00	14

L’interface SW (swath)

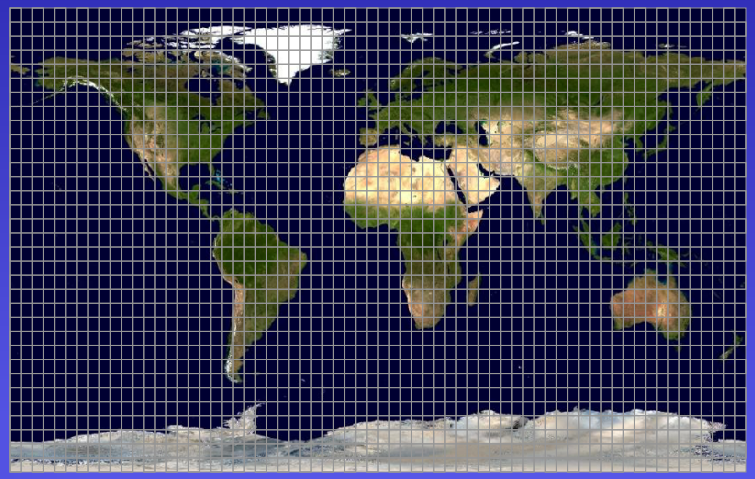
Support des données organisées chronologiquement (scans ou profils verticaux de satellites défilants par exemple).



L’interface GD (grid)

Support des données stockées dans un tableau basé sur une projection bien définie et supportée (6 familles prédéfinies).

données brutes + donneés de géolocation



projection : formule transformant un jeu de coordonnées terrestres en (ligne, colonne)

5. Outils associés

Utilitaires en ligne de commande

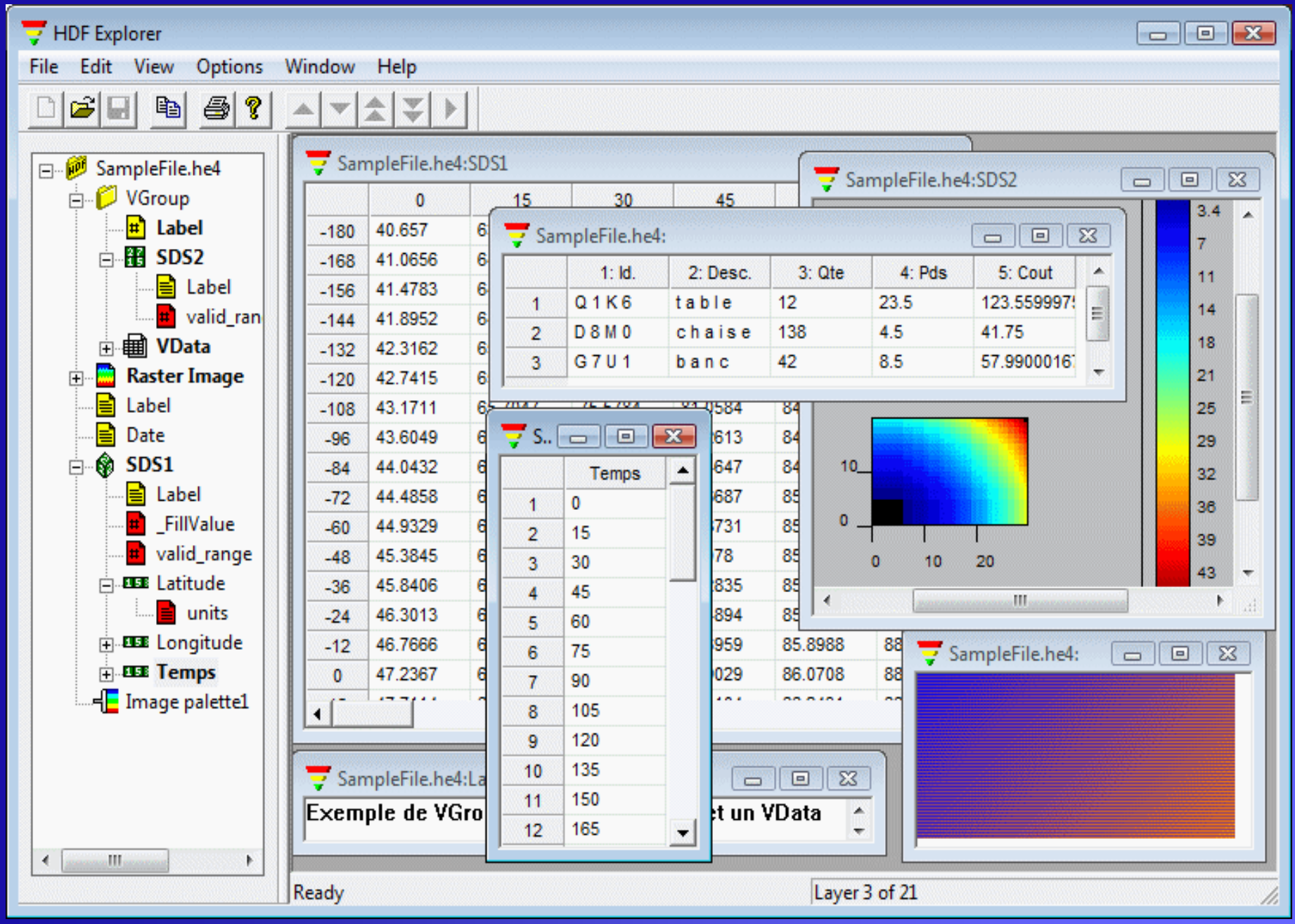
Il existe un certain nombre d'utilitaires en ligne de commande, fournis avec la bibliothèque, pour :

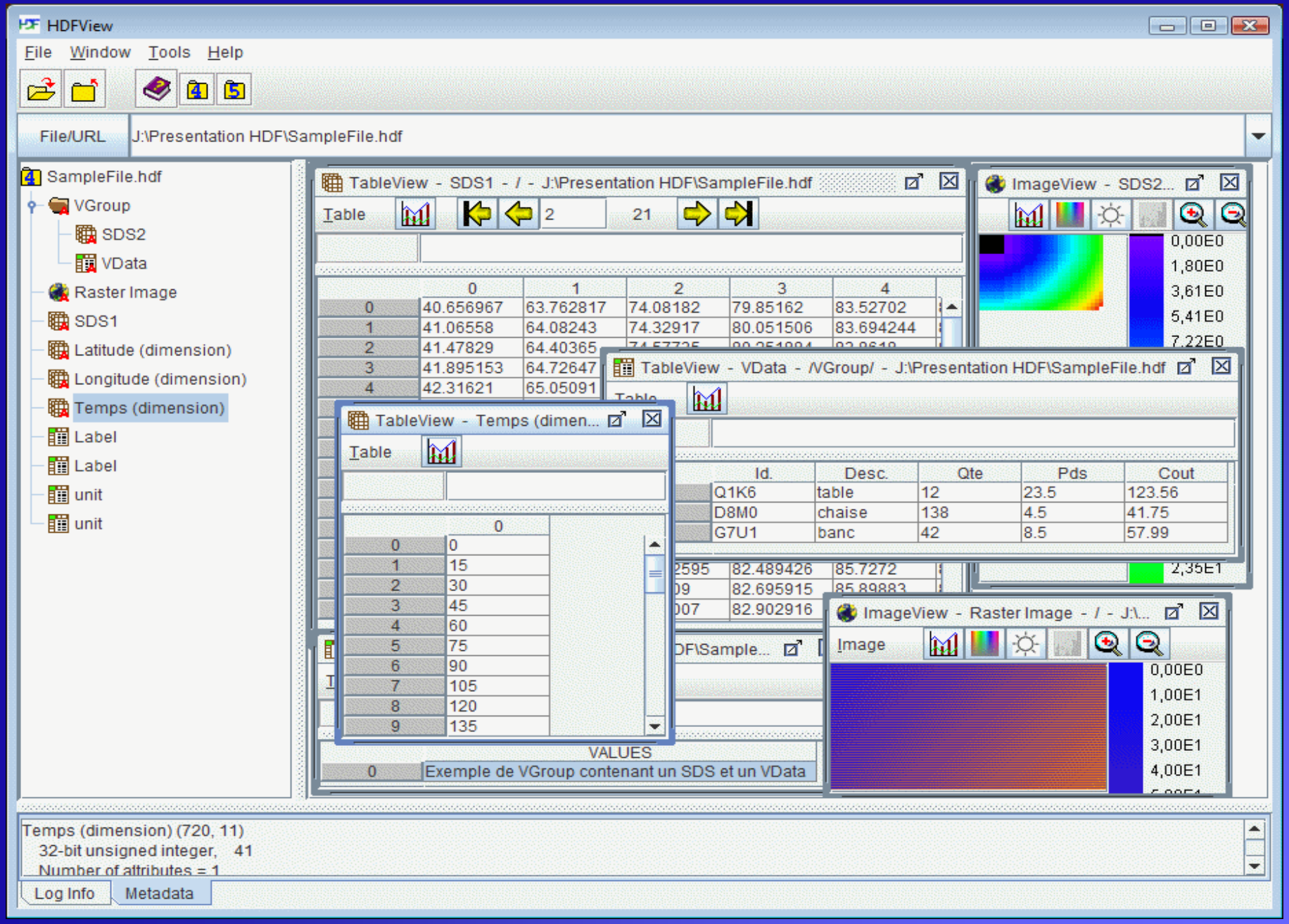
- Visualisation du contenu d'un fichier HDF ou de certains de ses éléments (*hdp*, *ncdump*)
- Comparaison de fichiers HDF (*hdiff*)
- Conversion de données brutes en HDF et inversement
- Conversion et compression d'images raster
- Conversion de HDF vers HDF5 (et de HDF5 vers HDF autant que possible)
- Compression de fichiers HDF
- …

Utilitaires réservés à un public averti !...

Exploration et visualisation

- HDF Explorer (Space Research Software)
  Windows
- HDFView (The HDF Group)
  Java – Toutes plates-formes
- HDFLook (LOA – Université de Lille 1)
  Unix - Linux
- …


HDF Explorer
File Edit View Options Window Help
SampleFile.he4
VGroup
Label
SDS2
Label
valid_ran
VData
Raster Image
Label
Date
SDS1
Label
_FillValue
valid_range
Latitude
units
Longitude
Temps
Image palette1
SampleFile.he4:SDS1
SampleFile.he4:SDS2
SampleFile.he4:
1: Id. 2: Desc. 3: Qte 4: Pds 5: Cout
1 Q1K6 table 12 23.5
2 D8M0 chaise 138 4.5 41.75
3 G7U1 banc 42 8.5
S..
Temps
SampleFile.he4:
Exemple de VGro
t un VData
Ready
Layer 3 of 21


HDFView
File Window Tools Help
File/URL J:\Presentation HDF\SampleFile.hdf
SampleFile.hdf
VGroup
SDS2
VData
Raster Image
SDS1
Latitude (dimension)
Longitude (dimension)
Temps (dimension)
Label
Label
unit
unit
TableView - SDS1 - / - J:\Presentation HDF\SampleFile.hdf
Table
2
21
0 1 2 3 4
0 40.656967 63.762817 74.08182 79.85162 83.52702
1 41.06558 64.08243 74.32917 80.051506 83.694244
2 41.47829 64.40365
3 41.895153 64.72647
4 42.31621 65.05091
ImageView - SDS2...
0,00E0
1,80E0
3,61E0
5,41E0
7,22E0
TableView - VData - /VGroup/ - J:\Presentation HDF\SampleFile.hdf
Id. Desc. Qte Pds Cout
Q1K6 table 12 23.5 123.56
D8M0 chaise 138 4.5 41.75
G7U1 banc 42 8.5 57.99
TableView - Temps (dimen...
Table
0
0
15
30
45
60
75
90
105
120
135
82.489426 85.7272
82.695915
82.902916
2,35E1
ImageView - Raster Image - / - J:\...
Image
0,00E0
1,00E1
2,00E1
3,00E1
4,00E1
VALUES
0 Exemple de VGroup contenant un SDS et un VData
Temps (dimension) (720, 11)
32-bit unsigned integer, 41
Number of attributes = 1
Log Info
Metadata

Développement

- C, C++, Fortran, Java, Python (SD, V, VS), …
- IDL (**I**nteractive **D**ata **L**anguage)
  Implémente la quasi-totalité des interfaces. Certaines limitations, par exemple concernant les attributs d'objet.
- MATLAB, Mathematica
- ENVI (**EN**vironment for **V**isualizing **I**mages)
- …

Liste des outils de manipulation et/ou de visualisation de données HDF : http://www.hdfgroup.org/tools.html

That's all folks !