



DIRK HOFFMANN, OLIVIER PISANO  
CENTRE DE PHYSIQUE DES PARTICULES  
DE MARSEILLE

# Pvss Data Viewer

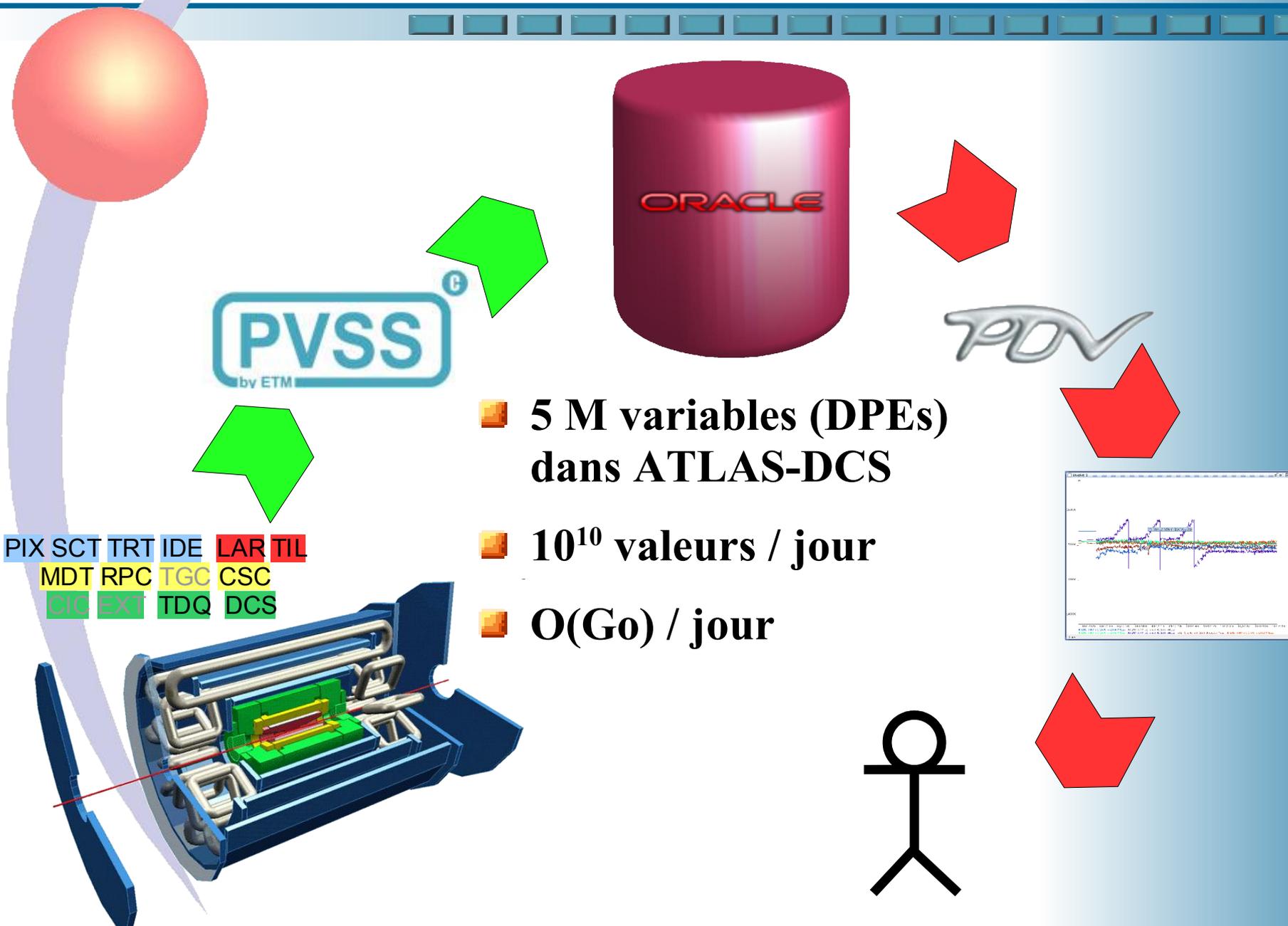
**Une application Java  
pour extraire et visualiser  
des données PVSS dans Oracle**

# *Plan de l'exposé*

- **Présentation de l'application PDV**
- **De l'idée à la réalisation**
  - Conception, cahier de charges
  - Choix techniques
  - Fonctionnalités actuelles / futures
- **Outils et environnements utilisés**
- **Résumé et perspectives**

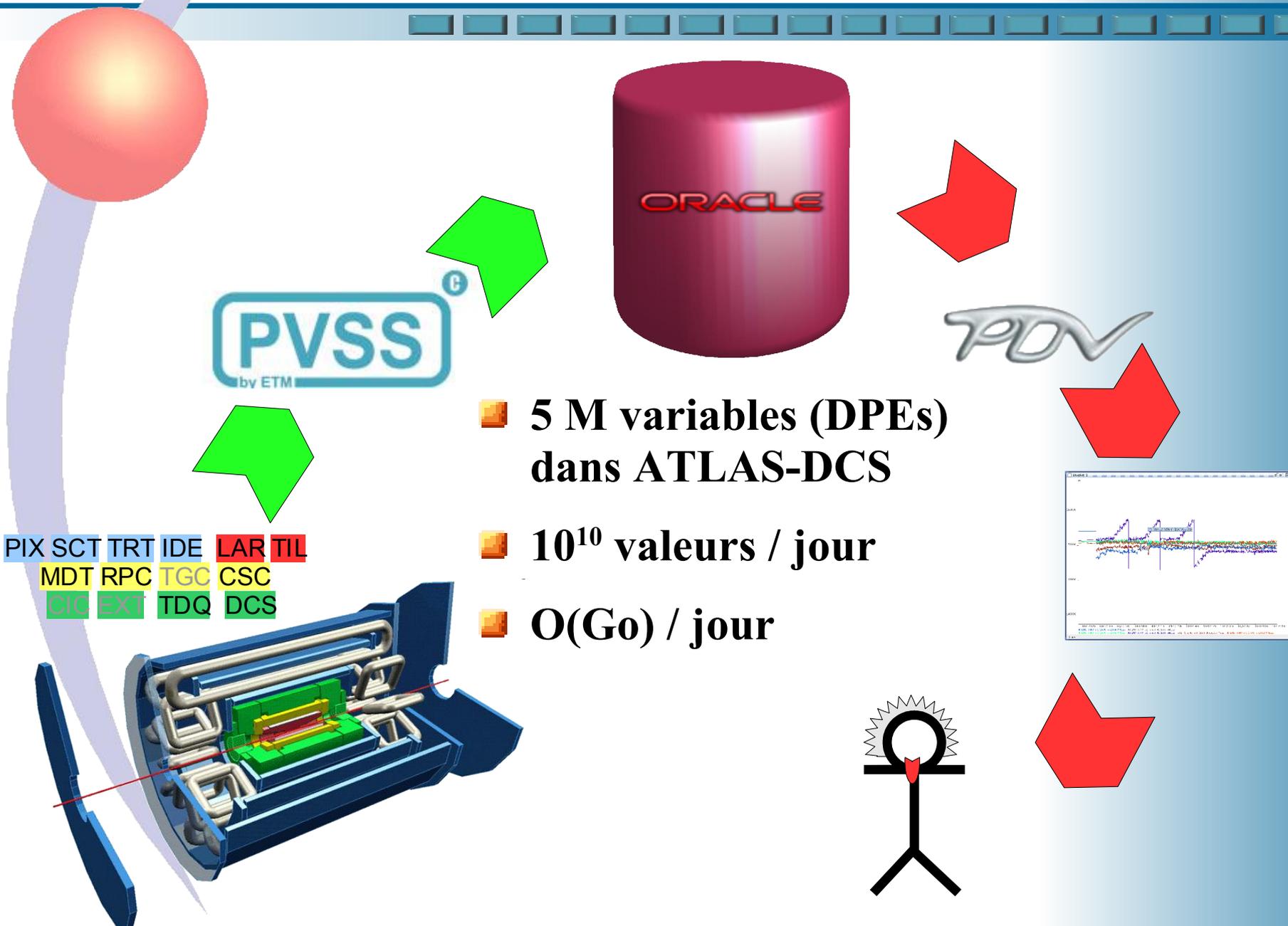
# Présentation du PDV

## Contexte



# Présentation du PDV

## Contexte



PIX	SCT	TRT	IDE	LAR	TIL
MDT	RPC	TGC	CSC		
CIC	EXT	TDQ	DCS		

- 5 M variables (DPEs) dans ATLAS-DCS
- $10^{10}$  valeurs / jour
- O(Go) / jour

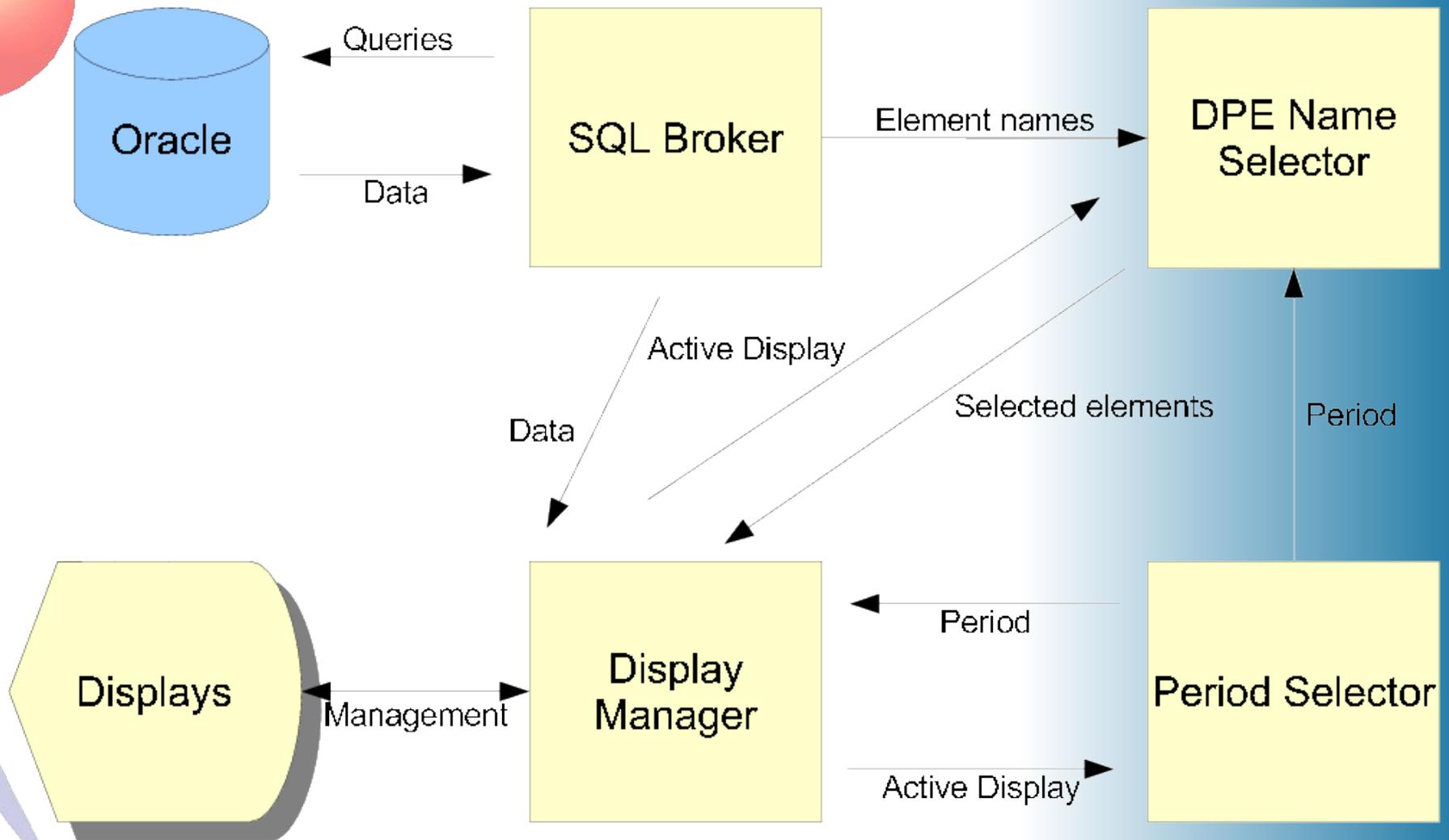
# *Présentation du PDV*

## ***Basic functions (design criteria)***

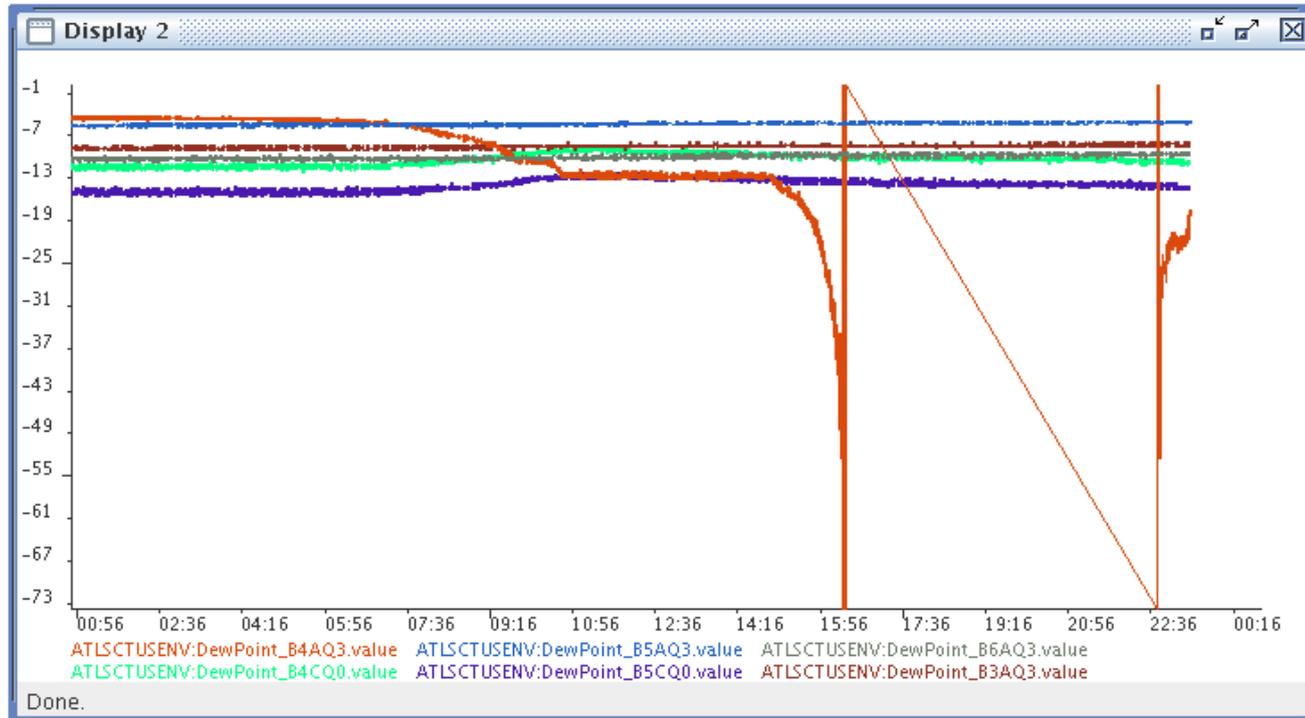
- **Universal (Java VM): no OS dependency, no environment/library dependency**
- **Retreive data from PvssDb, any schema, any PVSS RDB version (since 2006) from DPE name, alias, comment**
- **Efficient decompression and summary (min/max) for long-term displays ("two values per pixel")**
- **Export to "all" formats**
- **Plugin possibility (user specific code)**

# De l'idée au produit

## Conception (3 semaines)



# De l'idée au produit 1 semaine plus tard



Period Selector

From : 03/06/2007 00:49 Yesterday

To : 04/06/2007 00:49 Now OK

< - + >

Selected elements :

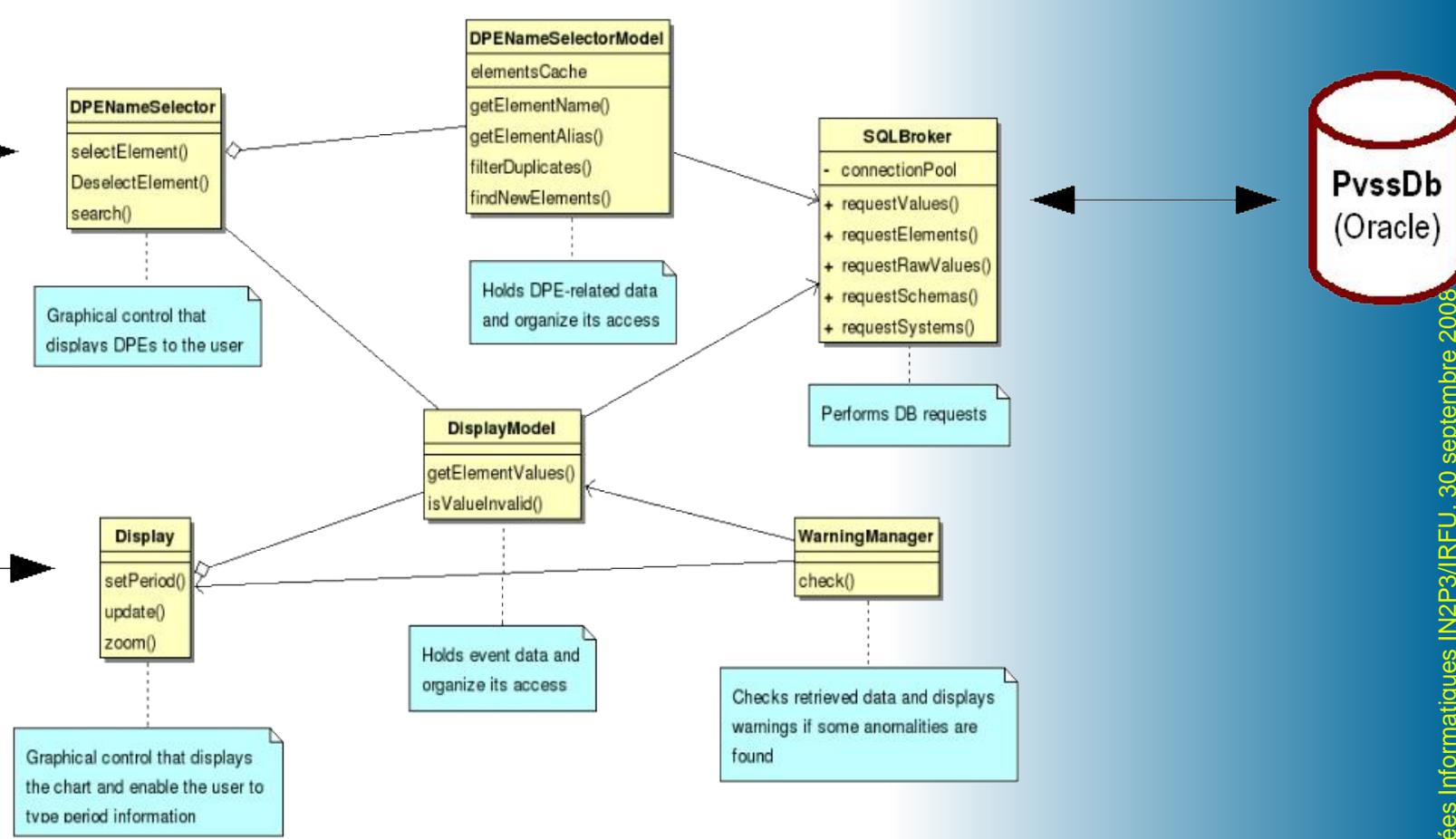
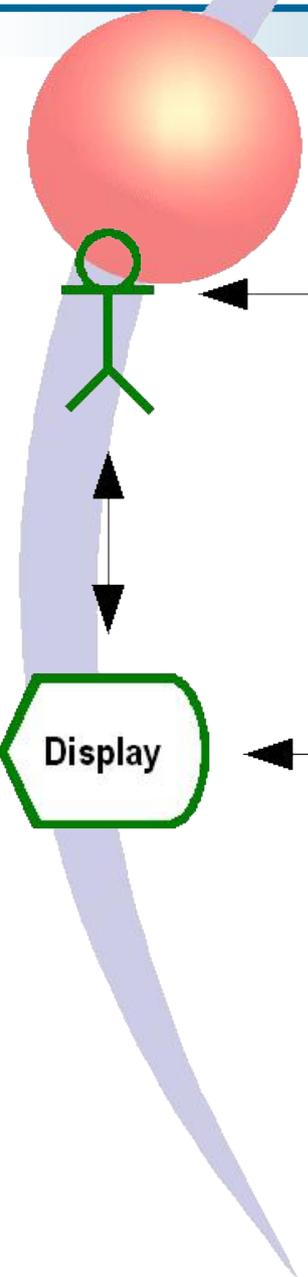
- ATLSCTUSENV:DewPoint
- ATLSCTUSENV:DewPoint
- ATLSCTUSENV:DewPoint
- ATLSCTUSENV:DewPoint
- ATLSCTUSENV:DewPoint
- ATLSCTUSENV:DewPoint

>> <<

ATLAS\_PVSS\_SCT Set OK

# De l'idée au produit

## 2 ans plus tard



# De l'idée au produit

## Tentative de valorisation

- PVSS est un produit commercial, largement utilisé dans le contrôle industriel (SCADA\*).



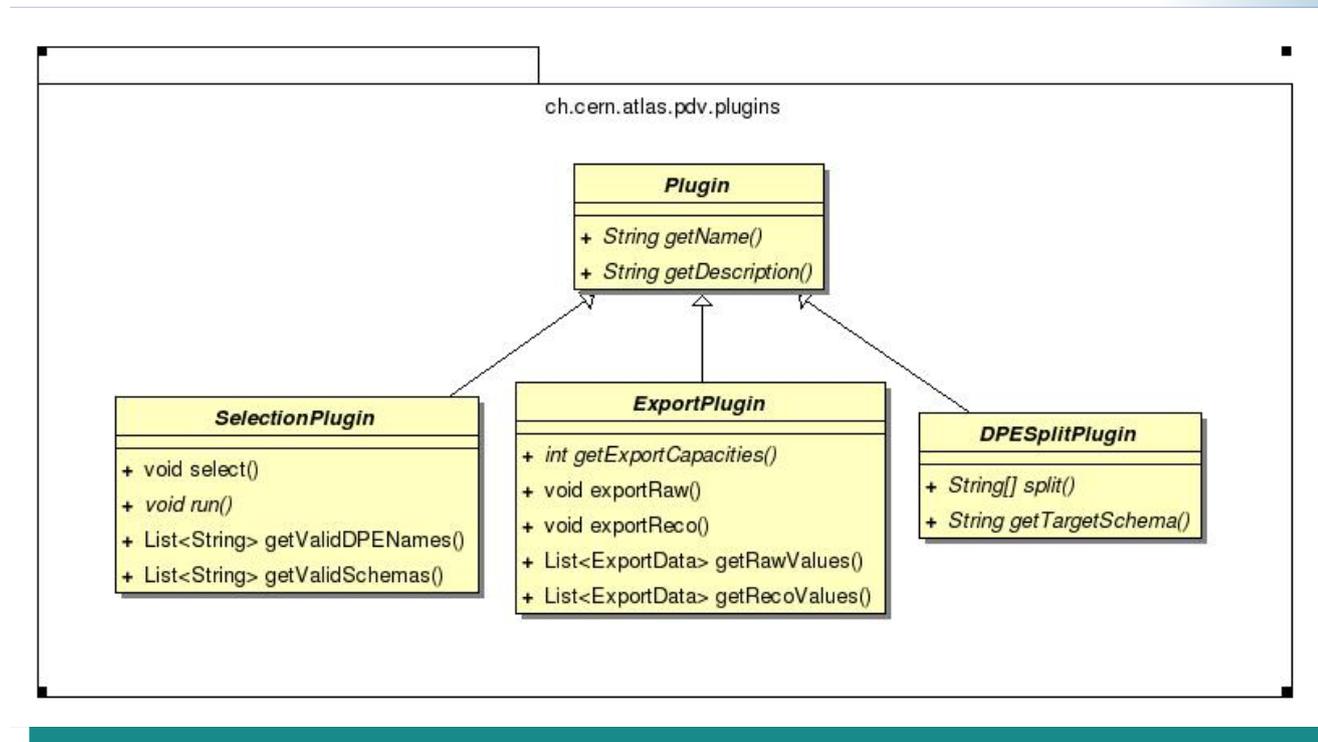
- Niveau de complexité trop élevé pour contrôle industriel
- Impossibilité de commercialiser à grande échelle (>CERN)

\*Supervisory control and data acquisition, „Slow Control“

# De l'idée au produit

## Equipe de développement «étendue»

- 2 personnes permanentes (quotité 40% + 100%)
- Contributions externes via greffons (*plugins*)



- 3 SelectionPlugins (3 sous-détecteurs)

# De l'idée au produit

## Choix techniques

### Langage de programmation

- **Java (JDK5, suite à contraintes CERN)**
  - Pré-compilation (*byte-code*)
  - Indépendance d'environnement et de système d'exploitation
  - Webstart par JNLP (Java Network Launch Protocol)
  - Compatibilité / universalité atteinte,
  - après quelques adaptations spécifiques.

# De l'idée au produit

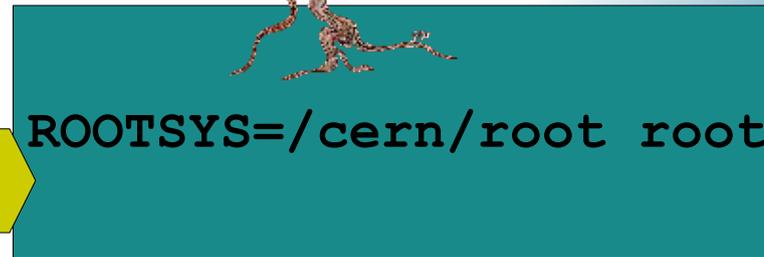
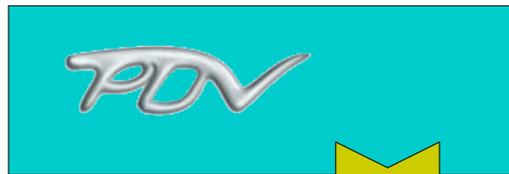
## Choix techniques

- **Librarie graphique** 
- **pas de critère particulier**
- **extrait de la description :**  
**JChart2D is intended for engineering tasks and not for presentations. It's speciality is run time - dynamic precise display of data with a minimal configuration overhead.**
- **1 développeur (D), à titre privé**
- **sourceforge.net**

# De l'idée au produit

## Choix techniques

- **Interface vers ROOT**

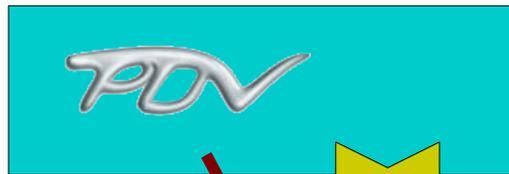


- **Absence d'écriture ROOT en Java (lecture, oui ! JAS)**

# De l'idée au produit

## Choix techniques

- Interface vers ROOT



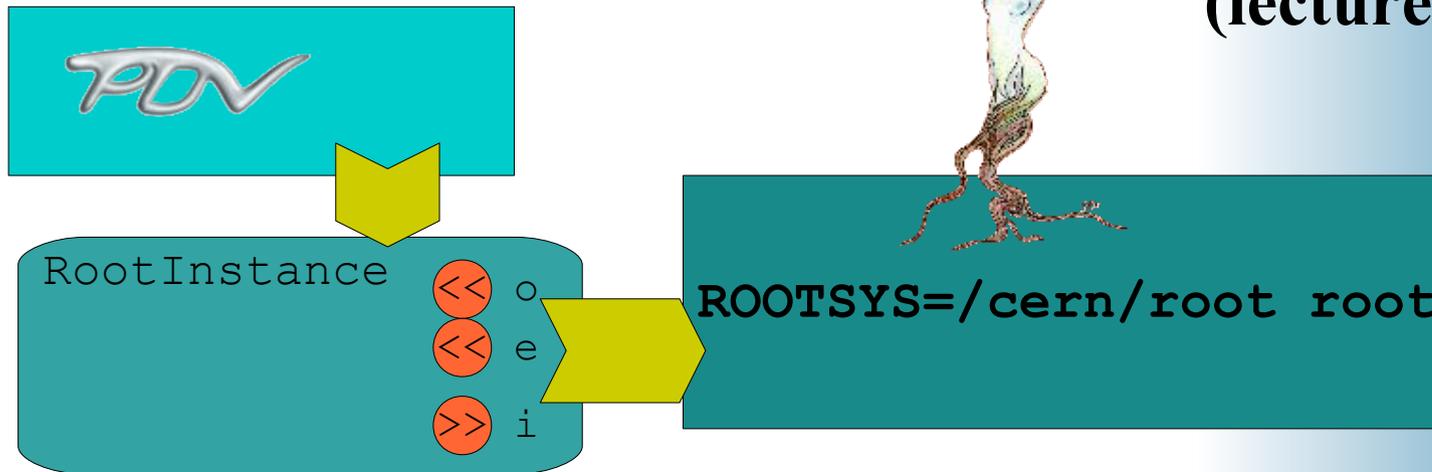
- Absence d'écriture ROOT en Java (lecture, oui ! JAS)

```
{  
  f = new TFile();  
  nt = new TNtuple();  
  nt->book();  
  nt->book();  
  // ...  
  nt->book();  
  f->close();  
}
```

# De l'idée au produit

## Choix techniques

- **Interface vers ROOT**



- **Absence d'écriture ROOT en Java (lecture, oui ! JAS)**

- **Suppose installation ROOT pré-existante.**
- **Pas de librairies natives (maintenance  $\times n$ )**
- **Indépendance OS conservée (en théorie)**

# Outils utilisés

## Eclipse

- **JDeveloper (Sun)**



- **Environnement intégré de programmation, écrit en Java, plugins C/C++, php, ...**
- **Gestion des classes, membres, packages**
- **Editeur syntactique**
- **Possibilités auto-édition : import, commentaires, modèles de fichiers**
- **Auto-compilation**
- **Test/debug graphique et intégré**
- **Excellent !**

# Outils utilisés

## Eclipse : Capture d'écran

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure for 'com.aramco.powers2.ui', including a 'test' directory containing 'NbBundleTest.java'. The main editor window shows the code for 'NbBundleTest.java', which includes imports, annotations, and test methods. The Console at the bottom shows the test results, indicating that the tests passed successfully.

```
27 import com.aramco.powers2.ui.NbBundle;
28
29 /**
30  * Tests the behavior of utility class NbBundle.
31  * Tests need to run against the background of a known set of objects.
32  * This set of objects is called a test fixture. (Refer to http://www.junit.org)
33  */
34  * @author Guanglin Du (dugl@petrochina.com.cn), Software Engineering Center, RIPED, PetroChina
35  */
36 public class NbBundleTest {
37
38     /**
39     * Uses the Bundle.properties to test NbBundle's behavior.
40     */
41     @Test
42     public void testExistingResource() {
43         String s1 = NbBundle.getMessage(ProjectView.class, "add_new_pvt_sat");
44         assertEquals("Add New PVT or SAT table", s1);
45     }
46
47     /**
48     * Uses the Bundle.properties to test NbBundle's behavior.
49     */
50     @Test
51     public void testNonExistingResource() {
52         String s1 = NbBundle.getMessage(ProjectView.class, "non-existing");
53         assertEquals("%non-existing", s1);
54     }
55
56     /**
57     * Method main to run this class directly.
58     * Can be run this way also on a command line:
59     * java org.junit.runner.JUnit4 samples.SimpleTestFixture
60     */
61     public static void main(String args[]) {
62         JUnitCore.main("com.aramco.powers2.ui.test.NbBundleTest");
63     }
64 }
65
```

Console Output:

```
Finished after 0.129 seconds
Runs: 2/2      Errors: 0      Failures: 0
com.aramco.powers2.ui.test.NbBundleTest [Runner: JUnit 4]
```

# Outils utilisés

## Eclipse : Quick Fix

The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows the project structure with packages `org.eclipse.banking` and `org.eclipse.banking.tests`. The `BankAccount` class is highlighted in the `org.eclipse.banking` package.
- Code Editor:** Displays the `BankAccountTests.java` file. The code includes an import for `java.math.BigDecimal` and a class `BankAccountTests` extending `TestCase`. It contains two test methods: `testDeposit()` and `testOverdraft()`. Both methods use `BankAccount` which is not yet imported.
- Quick Fix Menu:** A context menu is open over the `BankAccount` references. The first option is "Import 'BankAccount' (org.eclipse.banking)", which is highlighted. Other options include "Create class 'BankAccount'", "Create interface 'BankAccount'", "Change to 'BankAccountTests' (org.eclipse.banking.tests)", "Create enum 'BankAccount'", "Add type parameter 'BankAccount' to 'BankAccountTests'", "Rename in file (Ctrl+2, R direct access)", and "Add type parameter 'BankAccount' to 'testDeposit()'".
- Problems View:** Located at the bottom, it shows 6 errors and 1 warning. The errors are: "BankAccount cannot be resolved to a type" at lines 11, 19, and 27. The warning is: "BankAccount cannot be resolved to a type" at line 27.

```
package org.eclipse.banking.tests;

import java.math.BigDecimal;

public class BankAccountTests extends TestCase {
    public void testDeposit() throws Exception {
        BankAccount account = new BankAccount();

        assertEquals(new BigDecimal(900), account.getBalance());
    }

    public void testOverdraft() throws Exception {
        BankAccount account = new BankAccount();
        try {
            account.withdraw(new BigDecimal(100));
        } catch (Exception e) {
            // ...
        }
    }
}
```

# Outils utilisés

## Eclipse : Visual Editor

The screenshot displays the Eclipse IDE interface for editing a Java Swing window. The main editor shows a visual representation of a window titled "Information" with two text input fields labeled "Name:" and "Description:". Below the visual editor, the source code for the `getJTextArea()` method is visible, showing the logic for creating a new `JTextArea` if the current one is null.

```
private JTextArea getJTextArea() {  
    if (jTextArea == null) {  
        jTextArea = new JTextArea();  
    }  
    return jTextArea;  
}  
// @jve:decl-index=0:visual-constraint="10,10"
```

The interface includes several panels:

- Package Explorer:** Shows the project structure with `SwingApp`, `org.eclipse.swingapp`, `SwingFrame.java`, and `JRE System Library [SunJava2-1.4.2]`.
- Outline:** Shows the component hierarchy for the "Information" window, including `JContentPane`, `JLabel` (Name), `JTextField`, `JLabel` (Description), `JScrollPane`, and `JTextArea`.
- Properties:** A table showing the properties of the selected component.

Property	Value
background	Color:white
bounds	0,0,334,75
columns	0
componentOrientation	UNKNOWN
editable	true
...	...

The bottom status bar indicates the editor is `Writable`, `Smart Insert` is active, the zoom level is `125 : 1`, and the editor is `In Sync`.

# Outils utilisés

## CVS

- **Contrainte du CERN**  
(SVN pilote depuis les vacances)
  - **Gestion utilisateurs = AFS cern.ch**
  - **Auto-nightly build (pserver „anonymous“):**
    - javadoc
    - compilation
  - **CVS secondaire pour distribution des binaires !**  
(politique Atlas-DCS)
- *Outil classique, bien rodé,*  
*migration SVN à moyen terme*

# Outils utilisés

## Savannah



savannah.cern.ch

- Géré par CERN (projets des collaborations)
- Standard (seul) dans ATLAS
- Outil complet, configurable :
  - |                | <u>Bugs</u> | <u>Tasks</u> | <u>Support</u> |
|----------------|-------------|--------------|----------------|
| ■ Submit       | U E         | E            | U              |
| ■ Edit/Comment | U E         | U E          | E              |
  - Gestion des utilisateurs = AFS cern.ch
  - Configuration par fichiers souhaitable
  - *Utilisable, mais il doit bien y avoir mieux !*

# Outils utilisés

## Documentation

- **Différents niveaux**

- utilisateur
- développeur greffons
- développeur core

- **javadoc (fait partie de Java, donc Eclipse)**

- **TWiki**

- politique CERN / ATLAS
- gestion utilisateurs = AFS cern.ch
- Utile et suffisant pour documentation

mais limitations nb. pages et absence sous-structure !

# Outils utilisés

## Metrics

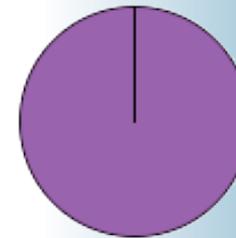
- Inscription d'un projet avec son CVS/SVN/git
- Analyse du code, par exemple :

-  [Mostly written in Java](#)
-  [Increasing year-over-year development activity](#)
-  [Well-commented source code](#)

[PVSS Data Viewer \(PDV\), updated 25 Sep 2008](#) 

### Languages

Java	100%
Other	
HTML	<1%
CSS	<1%
MetaFont	<1%



[PVSS Data Viewer \(PDV\), updated 25 Sep 2008](#)

[more at](#) 

### Project Cost

This calculator estimates how much it would cost to hire a team to write this project from scratch.

Include	<input type="text" value="Markup And Code"/>
Codebase	26,693 LOC
Effort (est.)	6 Person Years
Avg. Salary	\$ 55000 /year
<b>\$345,558</b>	

[PVSS Data Viewer \(PDV\), updated 25 Sep 2008](#)

[more at](#) 

# Outils utilisés

## Metrics 2

## PVSS Data Viewer (PDV)

Updated 25 Sep 2008 21:35 UTC

### Licenses

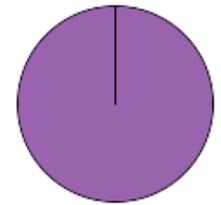
Ohloh searches the source code for individual license declarations. These licenses can differ from the project's official license.

[GNU Lesser General Public License 2.1](#) 135 files

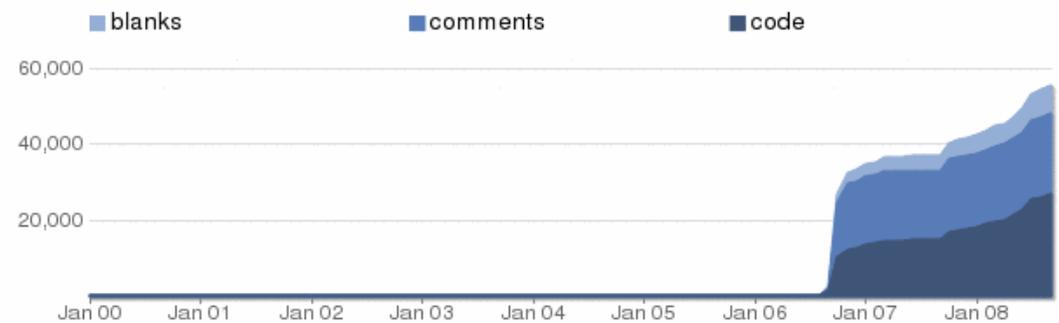
### Languages

Ohloh analyzes the project source code and determines the language of each line of code, excluding comments and blanks.

[Java](#) 100%  
[Other](#) <1%



### Codebase History

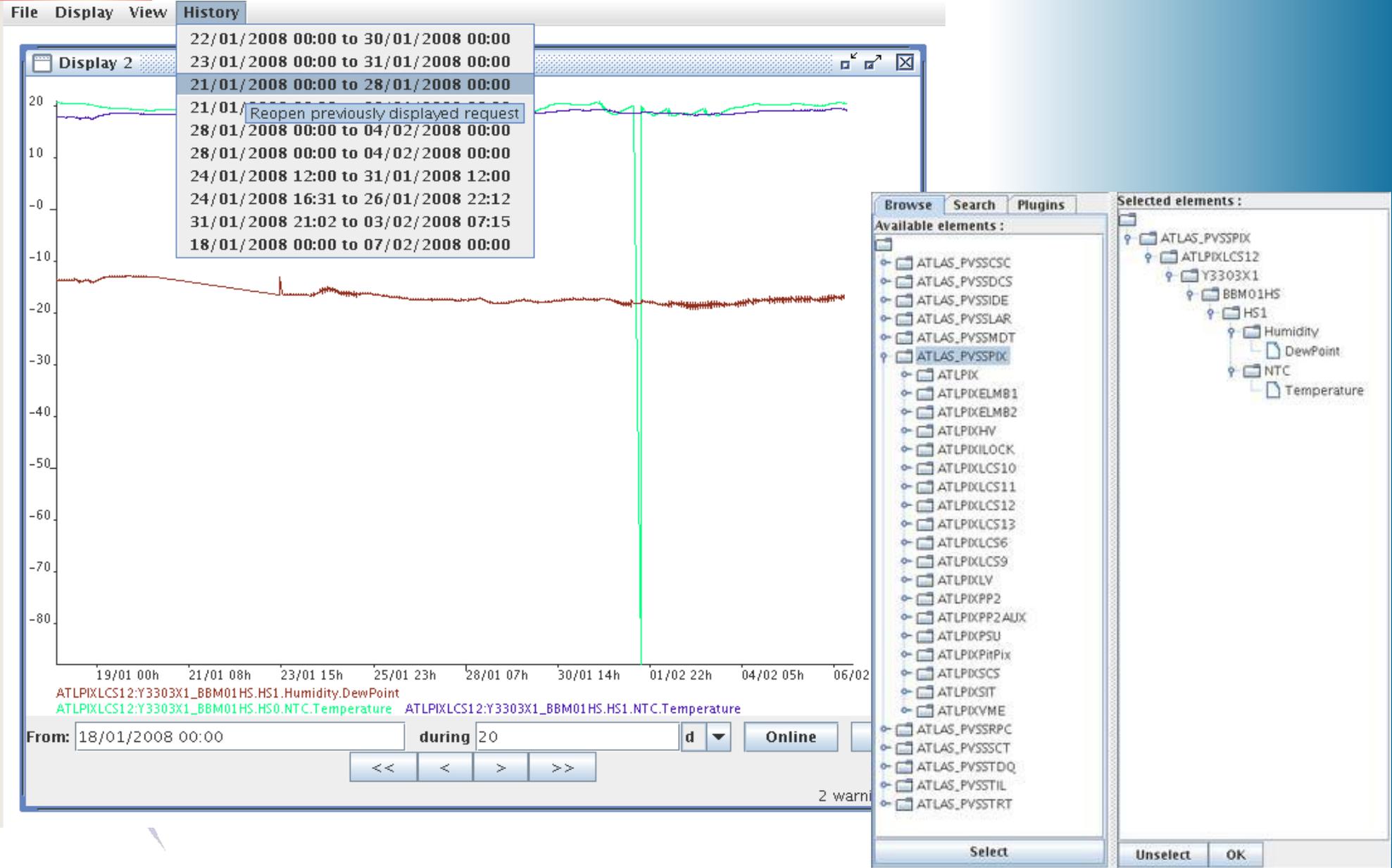


### Lines of Code By Language

Language	Code Lines	Comment Lines	Comment Ratio	Blank Lines	Total Lines
<a href="#">Java</a>	26,566	21,427	44.6%	7,186	55,179
<a href="#">HTML</a>	103	16	13.4%	40	159
<a href="#">CSS</a>	18	12	40.0%	8	38
<a href="#">MetaFont</a>	6	0	0.0%	0	6

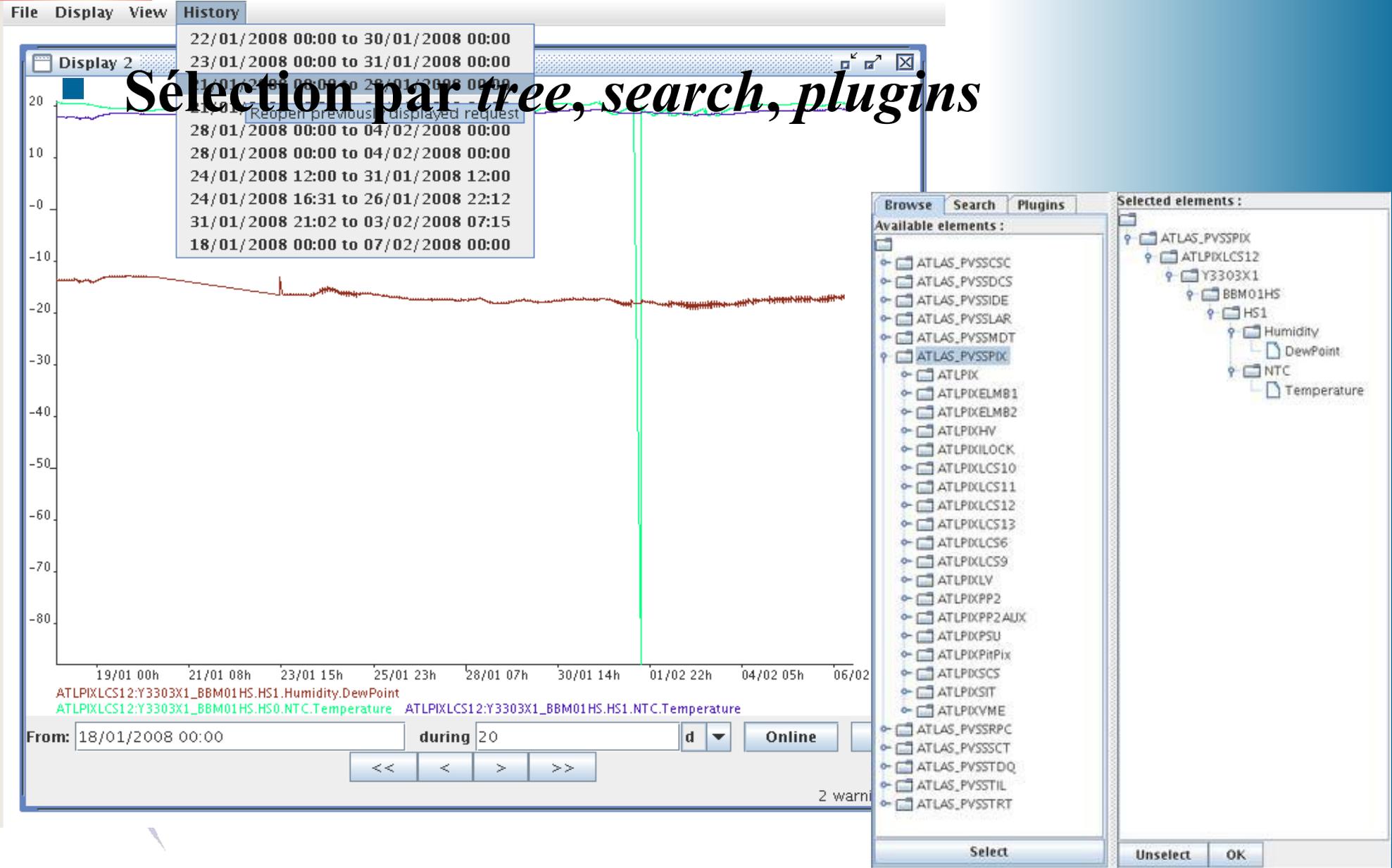
# Conclusion

## Etat actuel



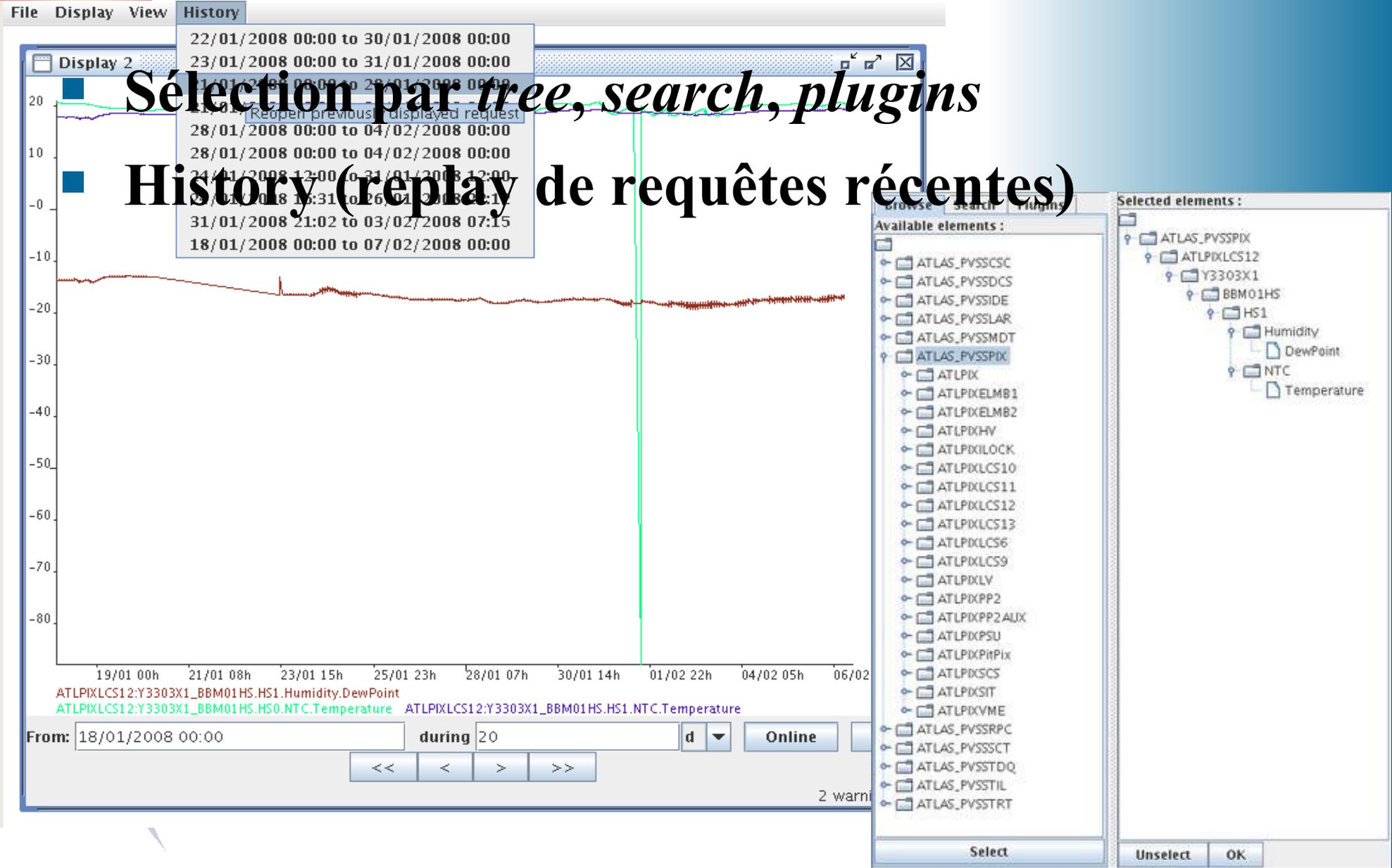
# Conclusion

## Etat actuel



# Conclusion

## Etat actuel

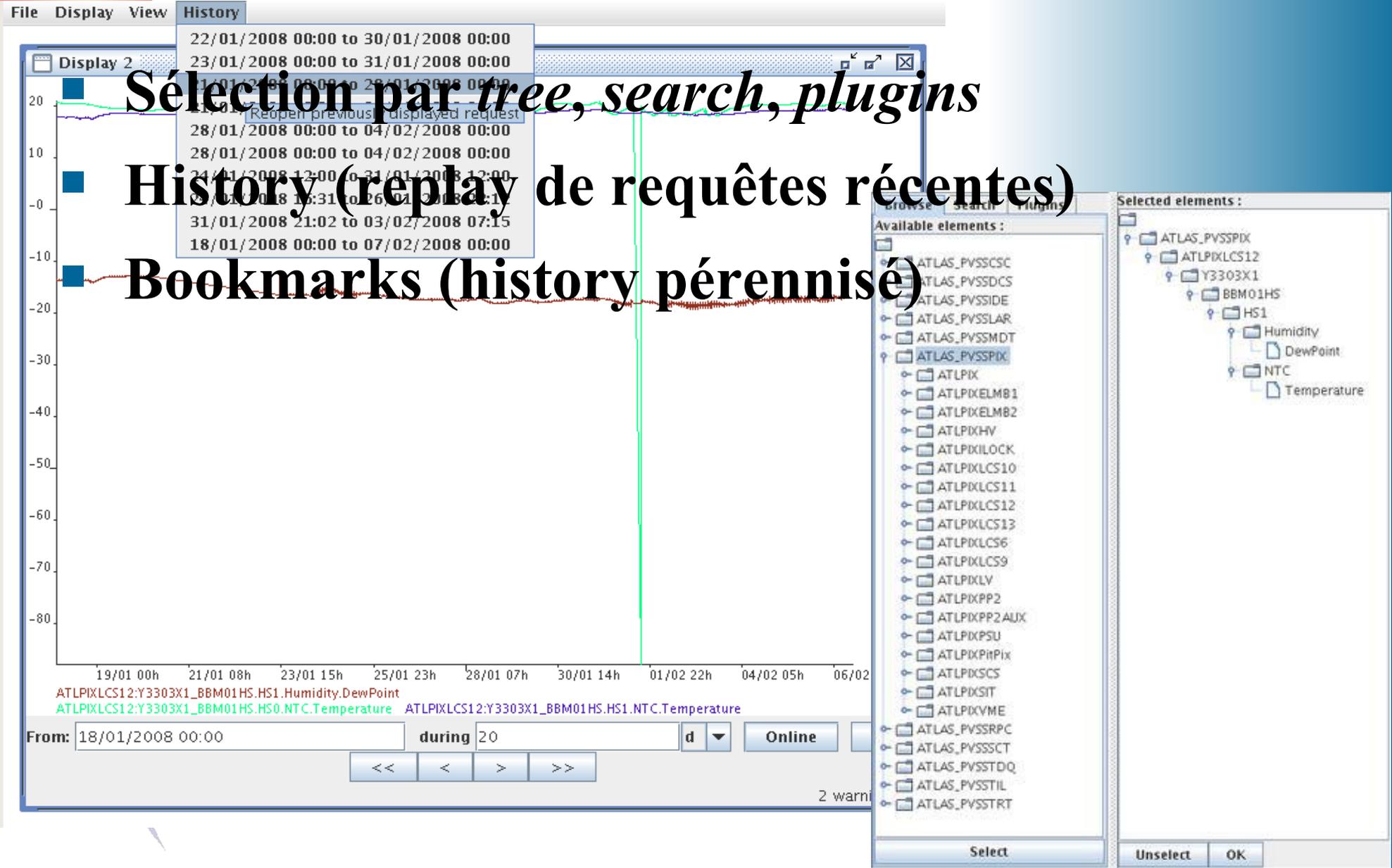


■ Sélection par *tree, search, plugins*

■ History (replay de requêtes récentes)

# Conclusion

## Etat actuel



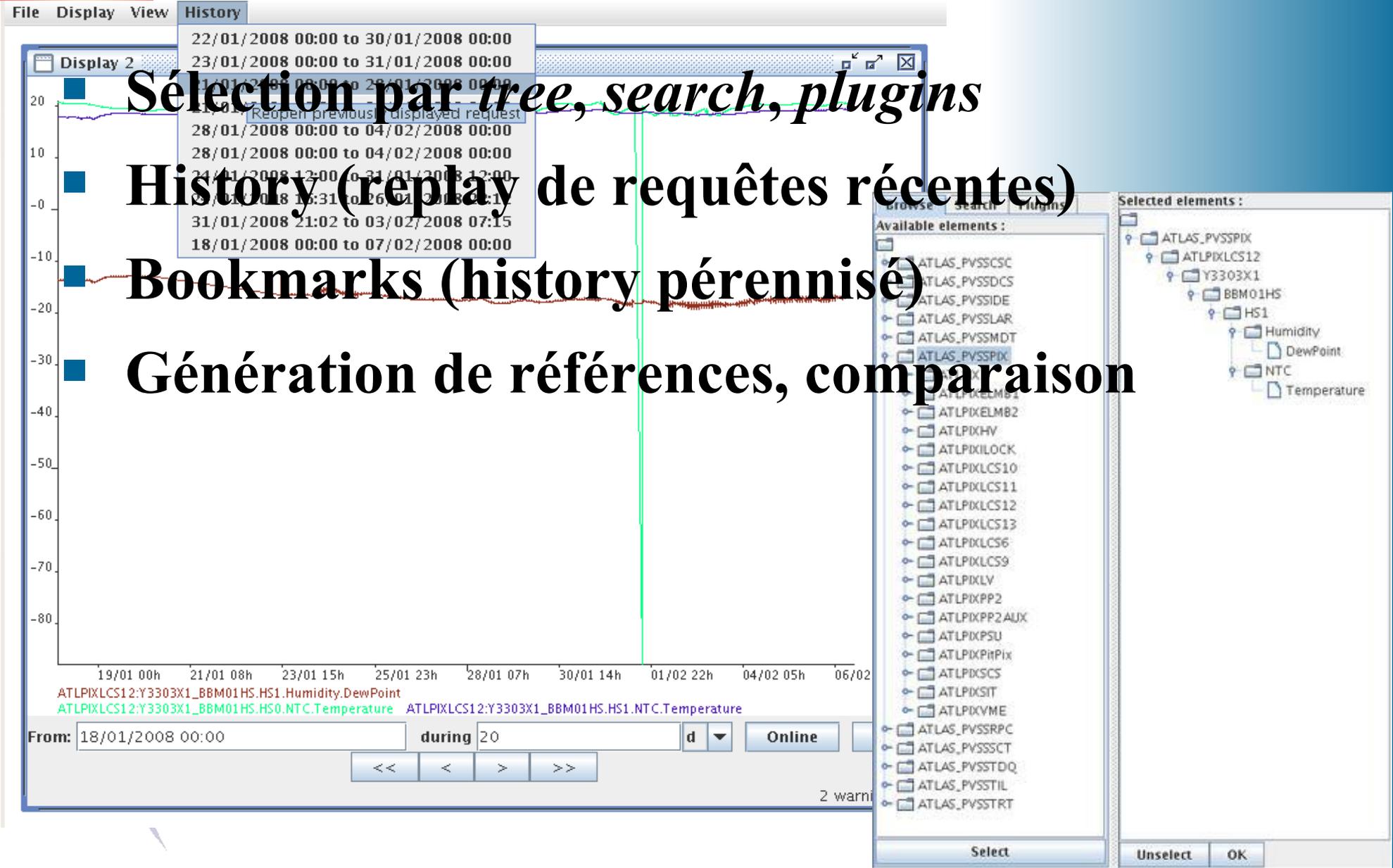
■ Sélection par *tree, search, plugins*

■ History (replay de requêtes récentes)

■ Bookmarks (history pérennisé)

# Conclusion

## Etat actuel



■ Sélection par *tree, search, plugins*

■ History (replay de requêtes récentes)

■ Bookmarks (history pérennisé)

■ Génération de références, comparaison

# *Conclusion*

## *Implemented features*

- **1D display, auto-update (timelines)**
- **Export to PNG, Printing,**
- **Webstart (Java Network Launch Protocol)**
- **Reactivity optimization: User history, Schema/PvssSystem buffering, pre-fetching, ...**
- **Plugins: Export, DpeSelector, DpeSplitter**
- **Retreive latest queries via query-history (XML, can be edited!)**
- **Resolves DPE name/alias history correctly!**

# *Conclusion*

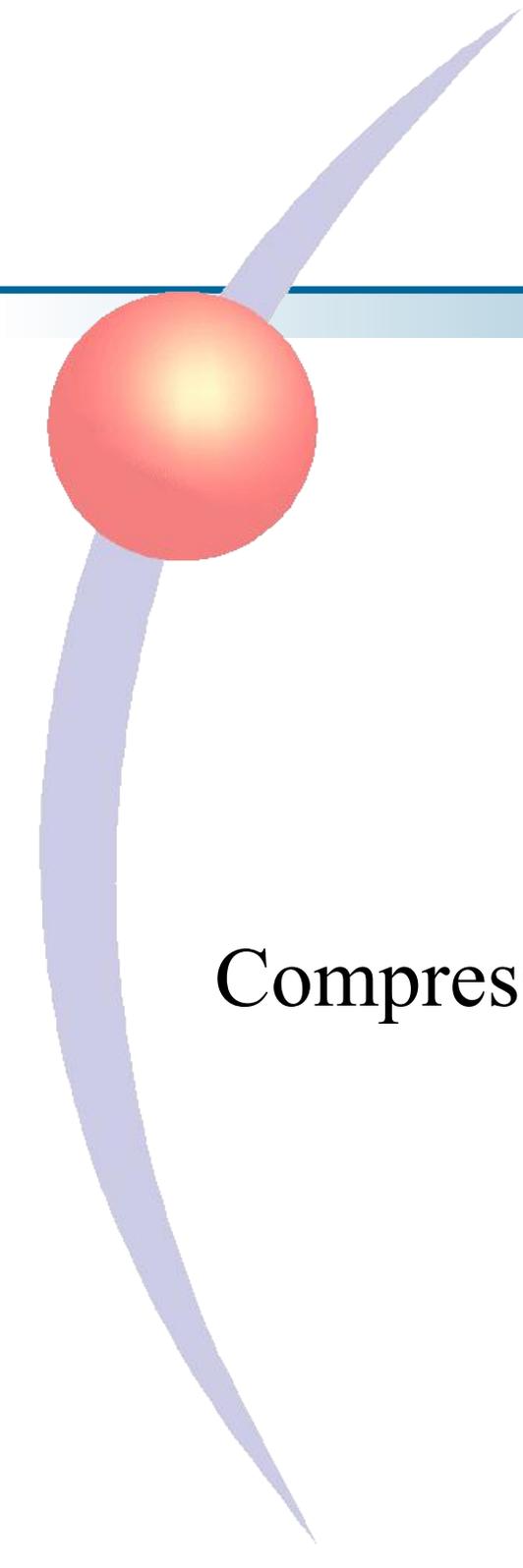
## **Perspectives**

- **Modifications / extensions de la librairie graphique**
  - projections (histogramme 1D)
  - corrélations (trace 2D)
  - règle de lecture pour valeurs
- **Pré-production de graphiques dans Oracle**
  - Intégration d'une partie du code PDV dans Oracle (Java stored procedures)
- **Accès sécurisé**

# Conclusion

## Résumé

- **PDV est une application graphique en Java**
  - qui extrait des données d'Oracle
  - et les visualise ou exporte dans une multitude de formats.
- **Il est devenu un outil standard dans Atlas, grâce à quelques choix techniques judicieux.**
- **Le projet complet comprend un ensemble d'outils, Eclipse, CVS, Savannah, Twiki, ...**
- **Expérience positive et excitante ; avenir brillant et pérenne possible.**

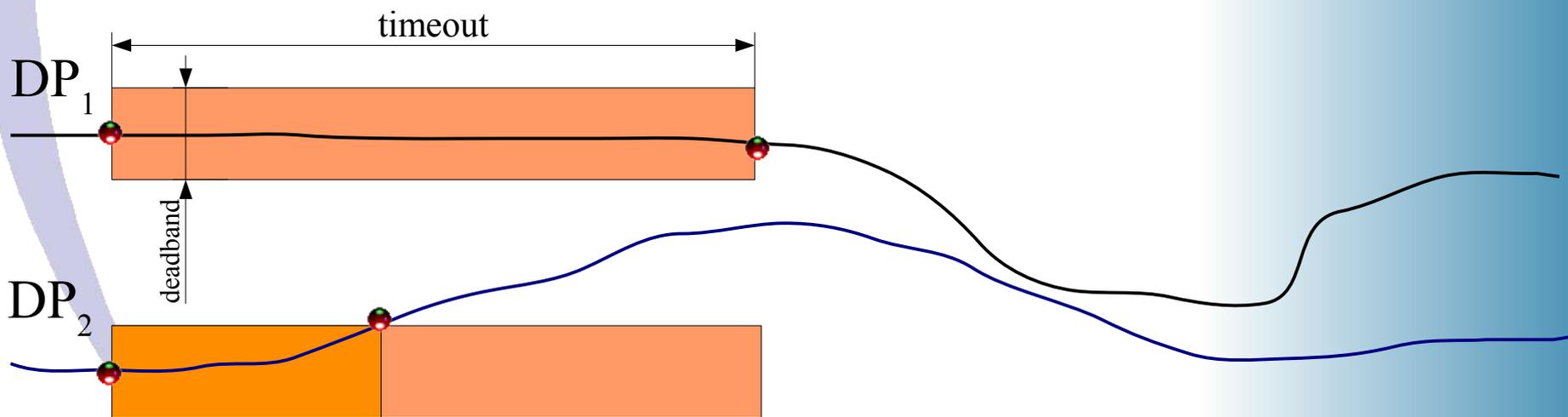


---

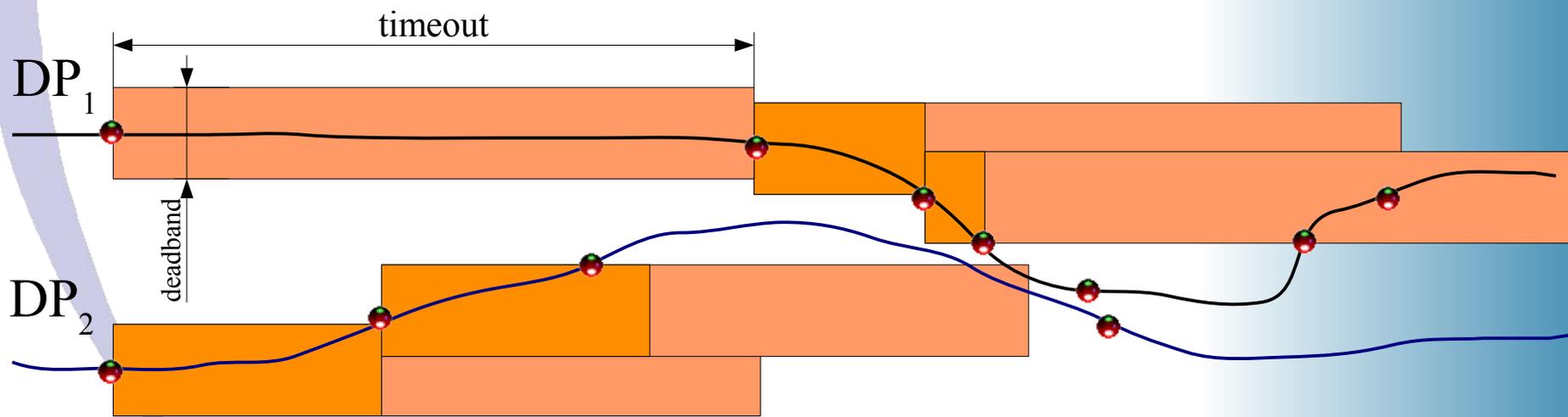
Réserve :  
Compression – Décompression – Récompression

# PvssDb data compression

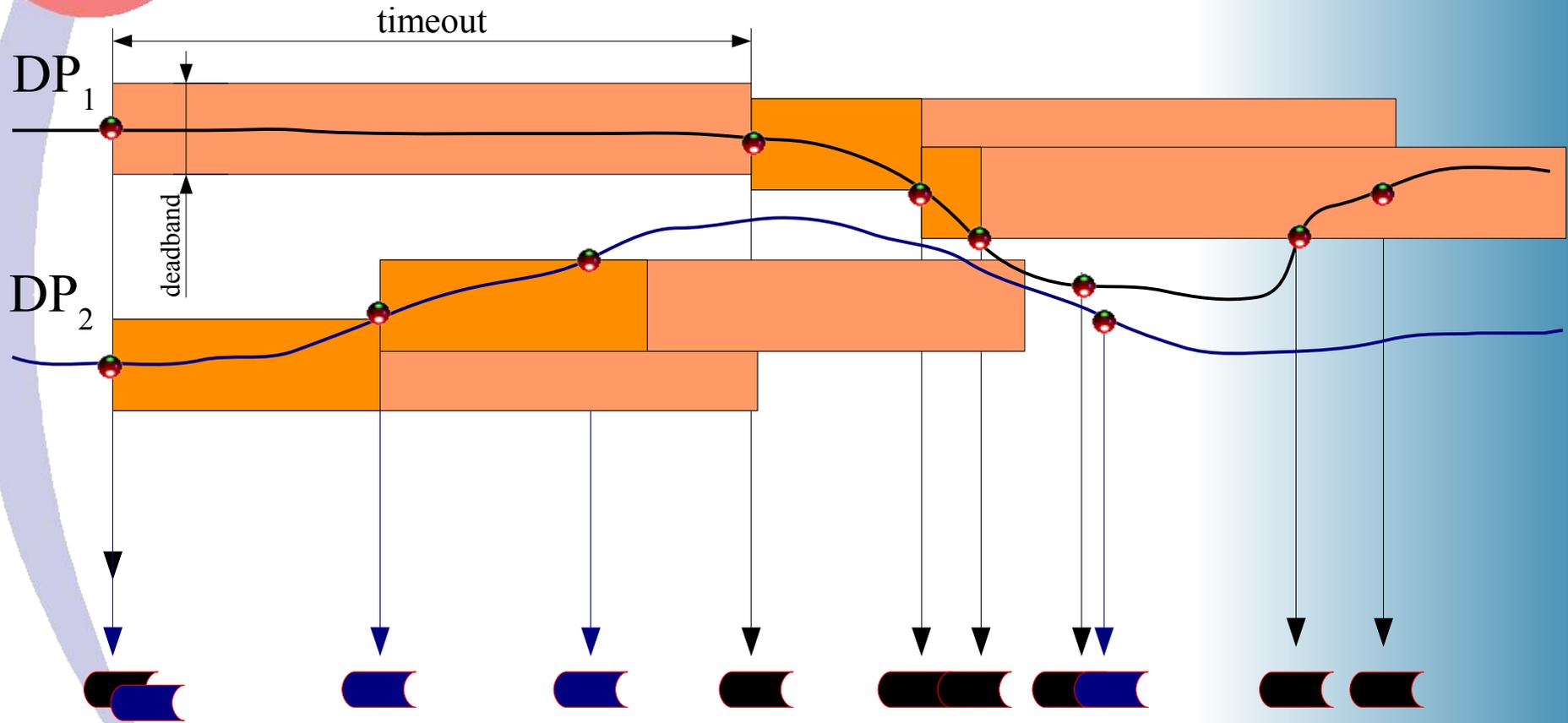
- **Incoming values recorded if**
  - deadband exceeded
  - timeout exceeded (resets deadband to zero)



# *PvssDb data compression*

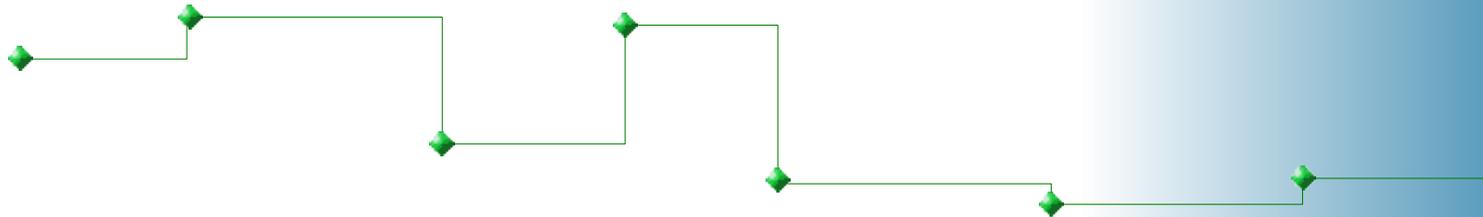


# PvssDb data compression



# Reconstruction from PvssDb

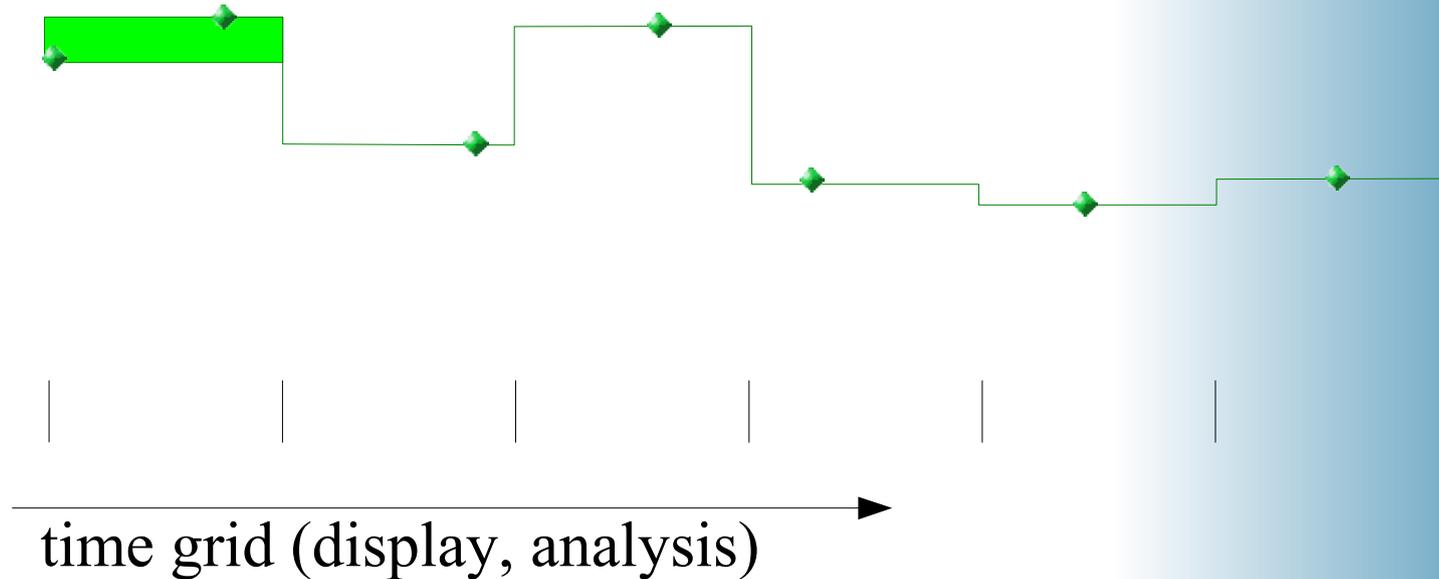
- **High Resolution** ( $t_{\text{disp}} \ll t_{\text{PVSS}}$ )



time grid (display, analysis) →

# Reconstruction from PvssDb

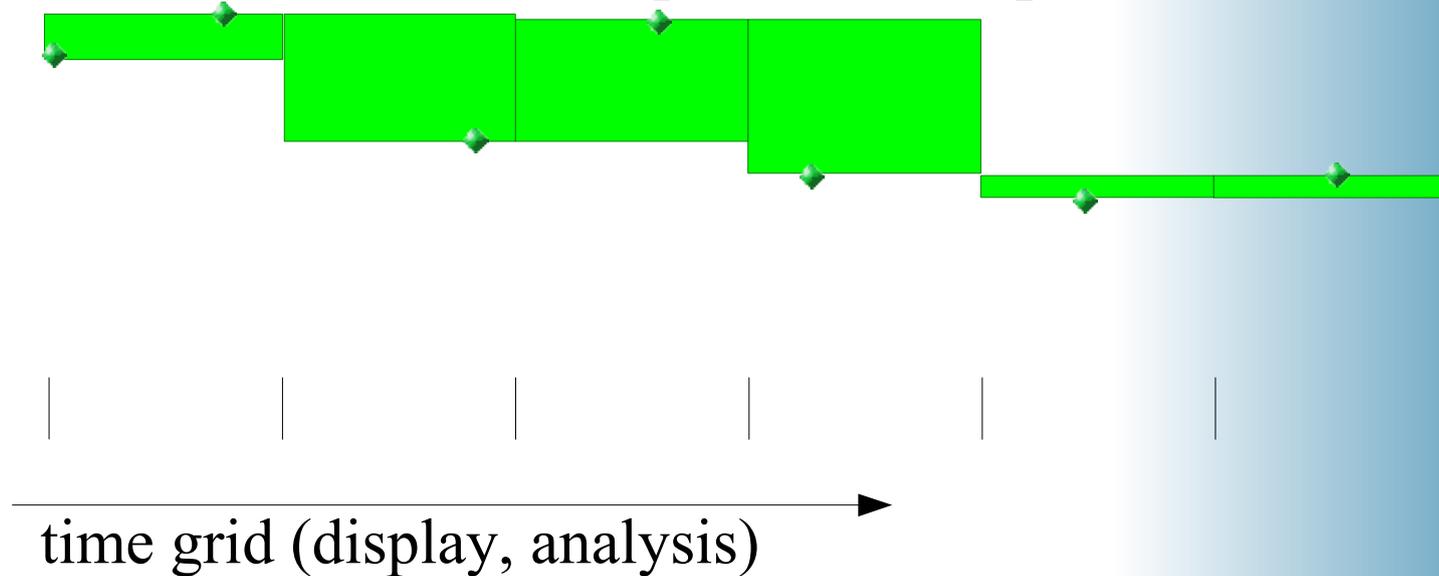
- **Low Resolution (or large time window)** ( $t_{\text{disp}} \gg t_{\text{PVSS}}$ )



# Reconstruction from PvssDb

- **Low Resolution (or large time window)** ( $t_{\text{disp}} \gg t_{\text{PVSS}}$ )

→ ... or rather like this? [MIN/MAX]



# Reconstruction from PvssDb

- **Low Resolution (or large time window)** ( $t_{\text{disp}} \gg t_{\text{PVSS}}$ )

→ ... or rather like this? [MIN/MAX]

