Panda Monitoring

by Dr. Valeri Fine

PAS / BNL

“Classic” Monitor

- Slow to render
- Unreliable (confusing) ;
- Hard to customize
- Hard to access the monitoring info for the custom processing
- Buggy (crashes too often)
- Impose unnecessary Db load
- Luck of the API documentation.

Classic" Monitor

- Slow to render
- Unreliable (confusing) ;
- Hard to customize
- Hard to access the monitoring info for the custom processing
- Buggy (crashes too often)
- Impose unnecessary Db load
- Luck of the API documentation.
- . . . .

RESOLVED !

Classic" Monitor

- Slow to render
- Unreliable (confusing) ;
- Hard to customize
- Hard to access the monitoring info for the custom processing
- Buggy (crashes too often)
- Impose unnecessary Db load
- Luck of the API documentation.
- . . . .

RESOLVED !

Mean time the other tools were evolving too.

From “Classic” to the “New Platform”

- The well-known “Classic Panda Monitor” I/F has been preserved.
- The “Classic Monitor” will not be deleted / removed. There is no technical reason to do that. I will be running “as is” albeit it will be “frozen”.
- If one knows how to use the old Monitor one already knows and can use the new one too.

New Monitor Project objectives

- The primary goal of the “new Monitor” project was not to change its look and feel rather to redesign its implementation to make it manageable.
- Separate the server and client code and provide the access the modern Web JavaScript / HTML5 tools – to provide the improve the existing Web client GUI.

Man power

Good:

- Time spent to design and implement the framework was about 2.5 months (~10% of my time for the last 2 years)
- 2-3 new feature requests / week to fulfill.

Bad:

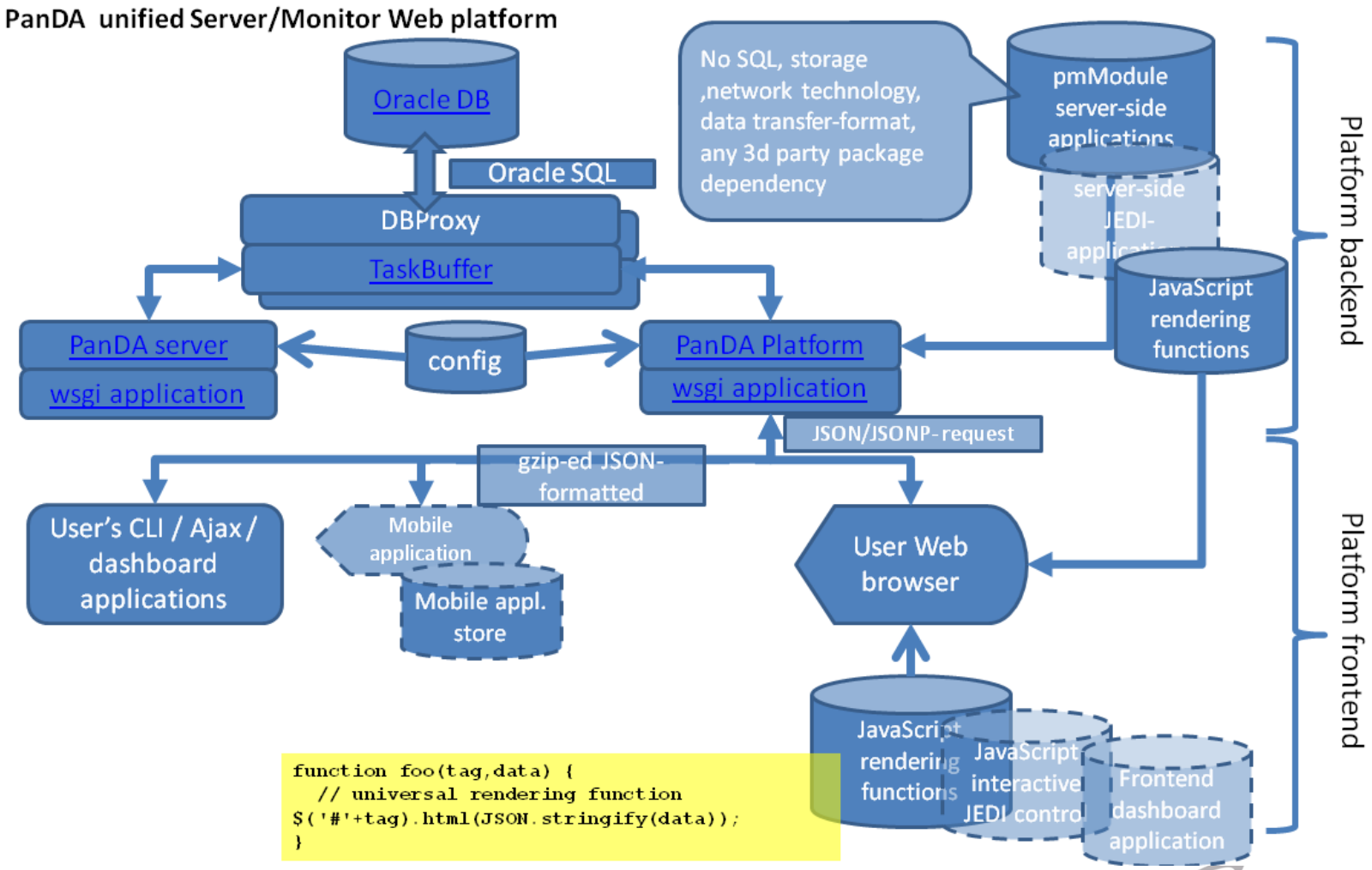
The “classic” Panda Monitor is about 80K lines from the 170 undocumented Python files. (10K to back prodsys-I)

To reproduce its functionality and its “look and feel” one has to read those files line by line with the naked eye to understand the logic and create the new platform compliant Python / JavaScript / Data-layer triad.

It is simpler to erect the new house from the scratch rather to rebuild the old one with the habitants keeping living inside.

New Panda Web Platform

https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/PandaPlatform



Monitoring goals:

- Identify exception
- Predict and avoid exception
- Gather the information to replicate (reproduce) the job flow if needed.

Easy to contribute: 3 components

To introduce the new Web application backed by the "Panda Web Platform" one needs to design and provide up to 3 software components, one mandatory and 2 optional

- **Python module (Panda plugin)** to publish the application Web API and the custom Web Content *(OO programming with "python" language is required)*

  Panda Plugin "Objectives"

  - Panda Plugin is the subclass of pmModule class to
    - **publish** one of its* own method* as "UI API" using **pmModule.publishUI** method (optional)
    - **create** some Python **object** and and **publish** it as the "Page Content" using **pmModule.publish**; method
    - Create and publish (optional) via **pmModule.publish** method the custom **JavaScript** function to render the "Page Content"
- (opt) **JavaScript function** to render the published Web Content *(programming with JavaScript, JQuery, and HTML are required)*
- (opt) **Data Access Layer** to fetch the external data used to generate the Web Content *(the required skill is defined by the nature of the data. For example, to fetch the data from Db one needs to know SQL)*

Panda Plugin API

- Any plugin can be invoked via either HTTP or HTTPS. CERN SSO is available also. No special httpd port configuration is required.

  The later means no special configuration to perform some “special” functions like “Logger” / “controller”. With the new platform the “logger” and “controller” both are the regular plugins.
- The Db schema components (column names) can be published as an element of the plugin API
- The time constrain API (days, hours, tstart, tend) can be added published as the components of API also

Web Platform API

```
http[s]://pandamon.cern.ch/[~version/][context/][application]
            [?param1=value1[&param2=value2[& . . . . [&paramN=valueN]
where
•   http: | https: - the network protocol
•   //pandamon.cern.ch - the platform Web Server host name
•   /~dev           - The framework version (optional)
•   /context        - the arbitrary application context name (optional)
•   /application    - the application name (optional) followed by the list of the non default
    parameter values

•   value<n>     - single value | regex pattern | comma-separated regex's
```

The framework support the arbitrary number of the concurrent versions. It allows to resolve the "Monitor version issue". Since we can provide the latest urgent corrections via the separate "dev" version and preserve the "stable tagged version" as well (For example: "pro" default version and the "dev" version)

For example:

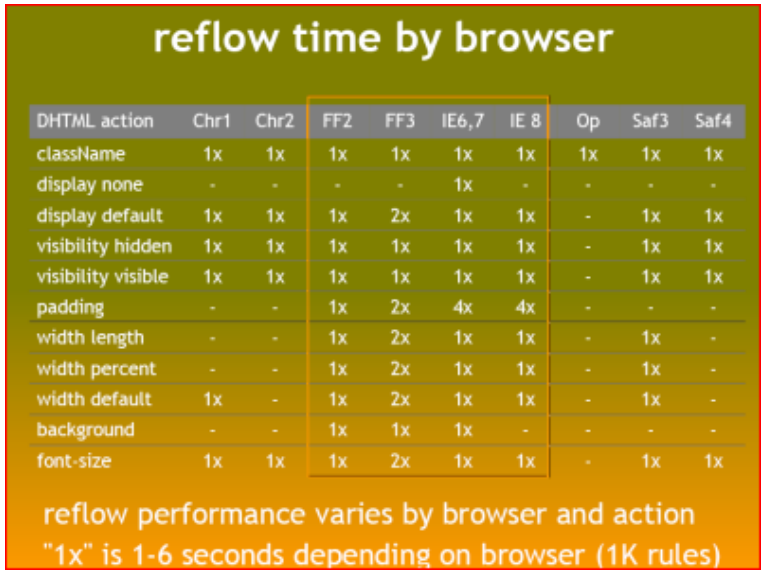
http://pandamon.cern.ch/jobinfo - stable tagged version

http://pandamon.cern.ch/~dev/jobinfo - the latest corrections and features

Monitoring Navigation

- Drill down (narrow search)
- Drill up (wider scope)
- Switch context

Watch for the browser reflow

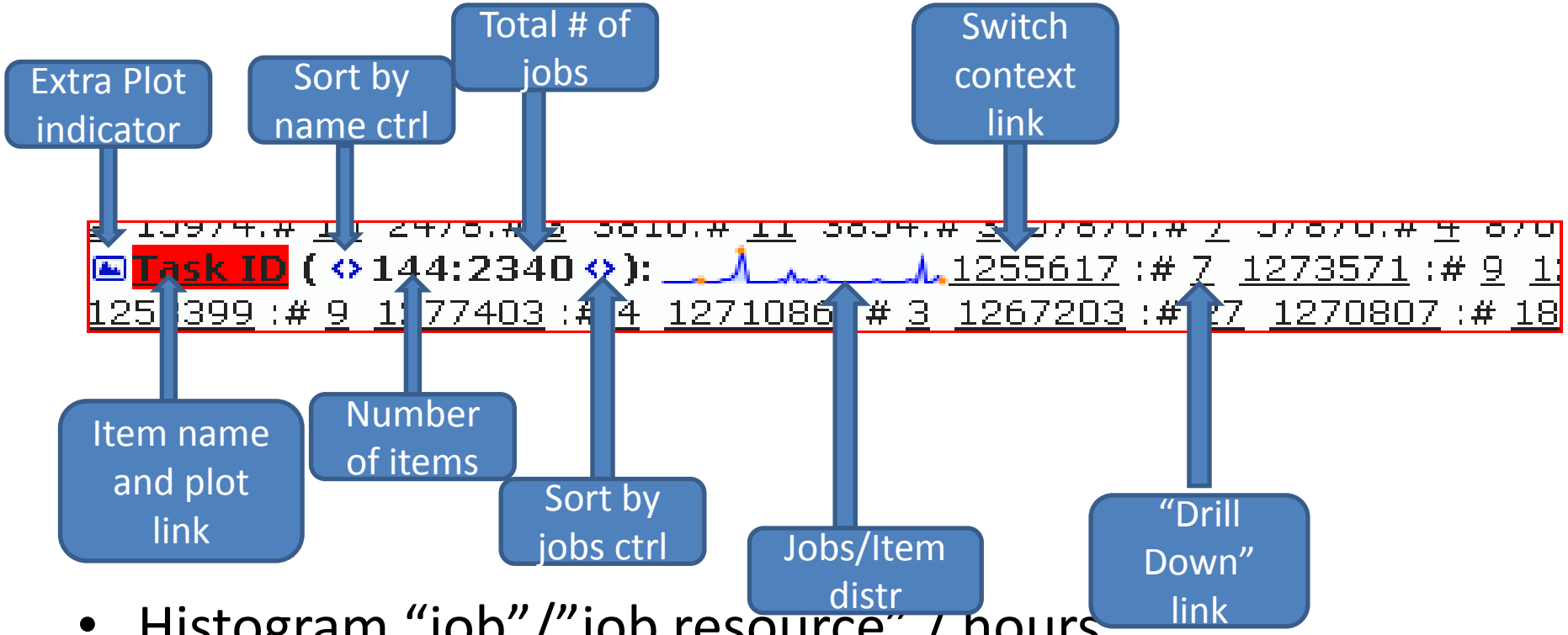


reflow time by browser

DHTML action	Chr1	Chr2	FF2	FF3	IE6,7	IE 8	Op	Saf3	Saf4
className	1x	1x	1x	1x	1x	1x	1x	1x	1x
display none	-	-	-	-	1x	-	-	-	-
display default	1x	1x	1x	2x	1x	1x	-	1x	1x
visibility hidden	1x	1x	1x	1x	1x	1x	-	1x	1x
visibility visible	1x	1x	1x	1x	1x	1x	-	1x	1x
padding	-	-	1x	2x	4x	4x	-	-	-
width length	-	-	1x	2x	1x	1x	-	1x	-
width percent	-	-	1x	2x	1x	1x	-	1x	-
width default	1x	-	1x	2x	1x	1x	-	1x	-
background	-	-	1x	1x	1x	-	-	-	-
font-size	1x	1x	1x	2x	1x	1x	-	1x	1x

reflow performance varies by browser and action
"1x" is 1-6 seconds depending on browser (1K rules)

- Select jobs: explicitly or via “drill up/down”, switch context GUI.



- Histogram “job”/”job resource” / hours

Histogram Package

The Python class PandaHistogram allows to create and publish the histogram objects. The client has the built-in capability to visualize it

Concurrency for JavaScripts

- Event though the JavaScript is the single thread Web application any Web browser does allow to fetch the data from the Web server concurrently.
- The feature is used to engage several server–side processes/ threads and nodes concurrently to build the single Web page content.

Data-layer and API parameters

Each ‘Panda Plugin' is free to define the format of the value it accepts. However, one can distinguish thee different formats, .

- **single value This is default format** The single value is the string defining the value that has no "," comma or '*' symbols. This is default format
- **regexp pattern** Slightly modified *regexp pattern* can be used to define the range of the possible parameter values.
  The symbol **dot** - "." is always just the ordinary character "." rather the special "pattern" for "any symbol".
  The symbol '*' is treated as the regexp expression '.*' **dot+asteric**, i.e. it is a "wildcard-like' pattern for "any number of any symbol"*.
- **comma-separated The list of the "comma-separated"** values. Each value can be either "single value" or "regexp pattern".

Data-layer and API parameters (cont.)

To handle all possible Platform API values the methods of the TaskBuffer sub-class have no hardcoded SQL queries.

The class methods use the special class ‘sqlSelect’ to generate the query.

It is to parse to the API values to generate the appropriated SQL “where” clause.

The “side effect” of such approach. To switch from one Db (Oracle) to another one (MySQL) it is enough to re-implement two methods of the said class: **`sqlSelect.vars()`** and **`sqlSelect.makeWhere()`**

Mobile applications.

The collaboration demand to provide the mobile monitoring application has to be foreseen now

- Full screen JavaScript/HTML5 application with no Web browser involved in between will become norm for the Linux desktop (for the Mac/ Windows and mobile devices it is already the norm)
- The current “Panda Platform” was designed to meet this new reality. However someone still has to write the code to take in account the mobile application feastures
- Limited bandwidth
- limited screen real estate space.
- The array of the different form-factors to fit in.
- A lot of CPU/GPU power to be enjoy.

Next

- File Info
- Pilot Info
- Dataset Info
- Summaries

The original plan reported during the last S&C Workshop was to complete the transition of the pages above by the end of the year (One should expect some 2-3 months delay due my Atlas-ROOT6 project participation)

Q/A ?

backups