Combined database

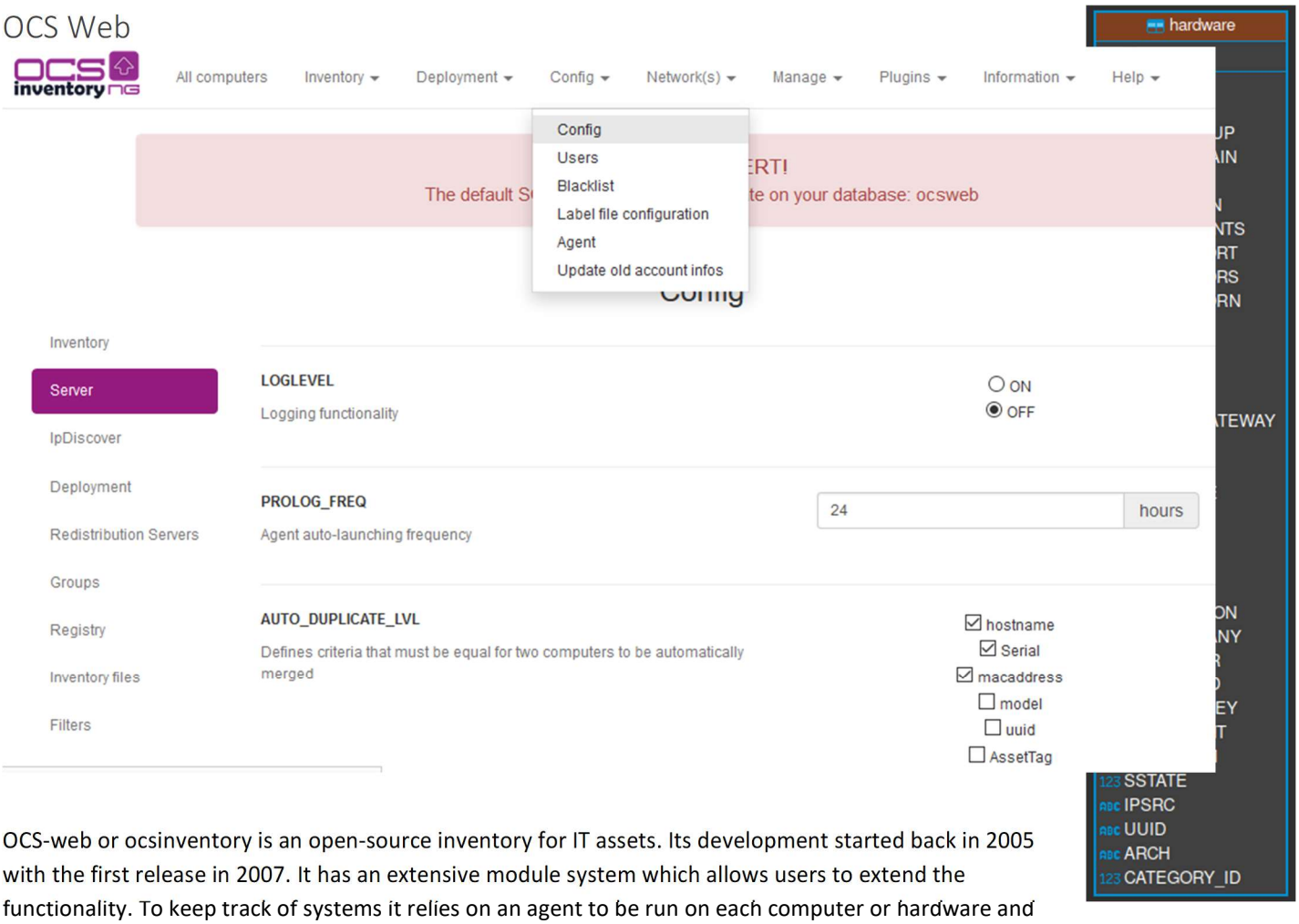
Situation

For our project it is important that we interface with the data that is gathered by Airbus. At this moment we do not have direct access to the logging and ticketing software that they use. As an intermediate we have received a copy of the databases that are generated by these software systems. In the following document I will go further into what each different software system is and which data it gathers in which way.

At this moment, the data of all 3 information gathering systems exists in 3 different databases. There is also no clear link between each computer throughout. There are some factors that will complicate the combination: each of the databases also has a different size and naming scheme and initial data is added manually which gives rise to errors in data.

It is important that we find an easy to understand and logical approach to combining the 3 databases for us to be able to implement the combined dashboard that we are developing. Later, this same database will be used by an AI and this will hopefully lead to a predictive maintenance system.

For each software system I will give a short history afterwards I will dive deeper into how a computer is represented and finally which data is relevant to us and how it is stored in the database.



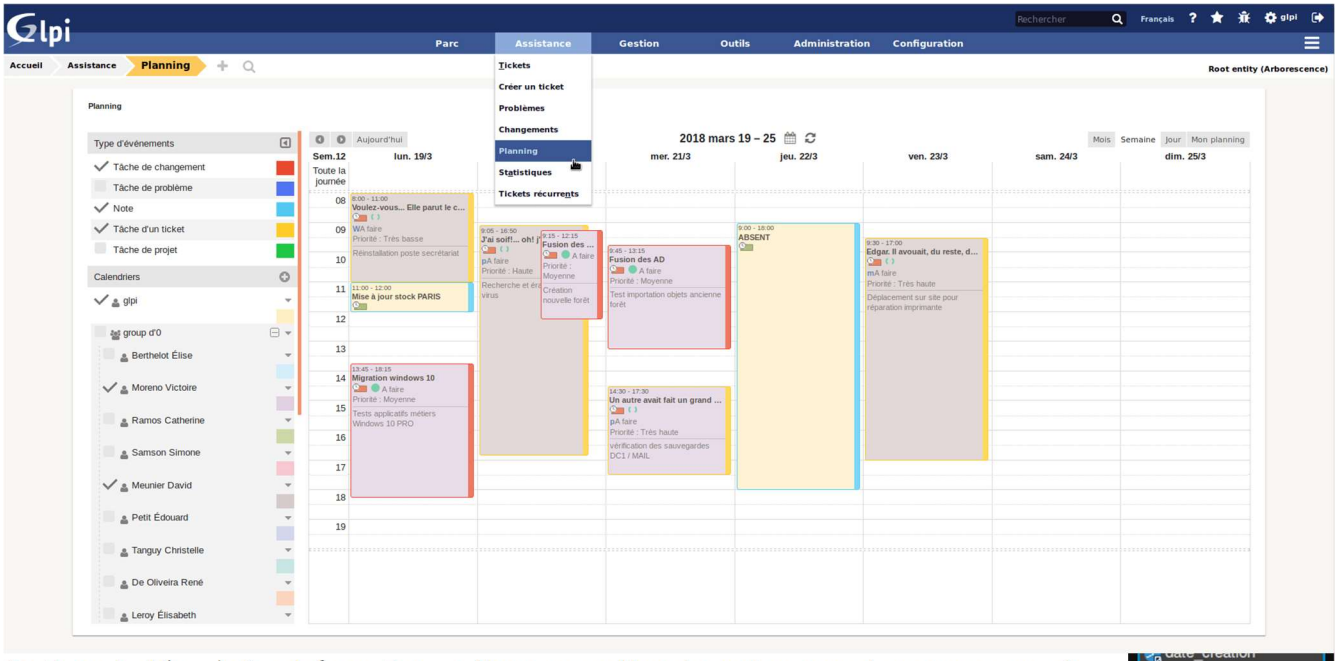
OCS-web or ocsinventory is an open-source inventory for IT assets. Its development started back in 2005 with the first release in 2007. It has an extensive module system which allows users to extend the functionality. To keep track of systems it relies on an agent to be run on each computer or hardware and supports all major operating systems from servers to personal computers and even Android phones. The underlying technologies are MySQL, Perl, and Apache.

This is the smallest of the 3 database systems and contains around 50Mb of data. The scale of rows is from 1 000 to 10 000 for the relevant rows.

All computers and systems are kept in a table named “hardware”. Each computer is assigned an id, a deviceid and a name. In this table they also keep basic information about the hardware of the system and which operating system it is running.

The most relevant table we need from OCS are the software installs. These are kept in the table Software. Each installation is linked back a computer through the id from the hardware table. For each installed software it has facilities to keep track of install date, name, publisher, version, folder, comments, filename, file-size, source, install date. Most of the time only the software name, publisher and install date are available.

GLPI



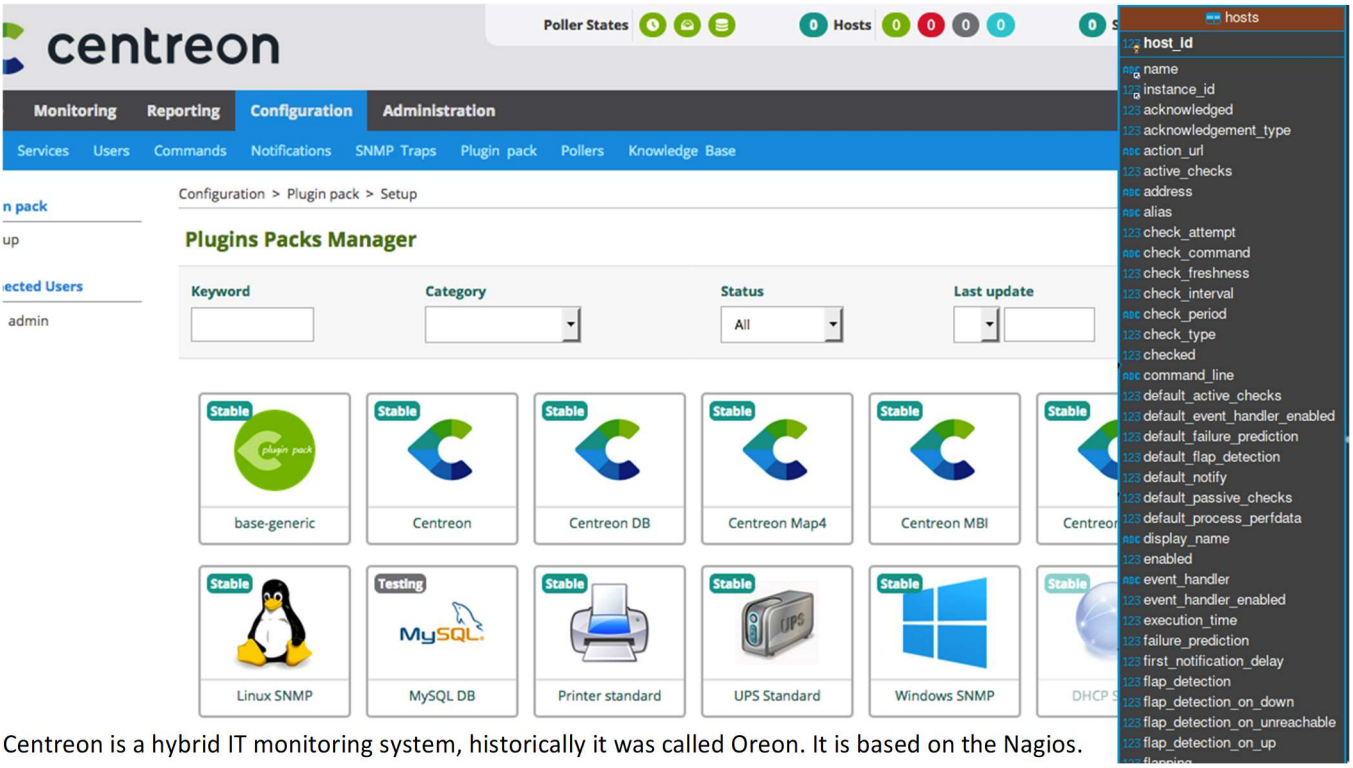
Gestionnaire Libre de Parc Informatique or Open source IT equipment manager is an open source is an open source IT asset manager, service desk and issue tracking system. It has been in development since 2003. It is reliant on partners for development and financial support. The main developer at this moment is Teclib' but all the code still falls under the GPL license. The software itself is mainly written in PHP with a MySQL database back-end.

The database itself takes around 84Mb when and has a scale of 10 000 – 100 000 rows that are relevant to our application.

Computers are kept in the glpi_computers table. It is assigned a name, serial, otherserial and uuid to identify the computer in the system. From this database we are interested in getting out the tickets. Tickets are kept in the glpi_tickets table and follow ups are kept in glpi_followuptickets. These follow ups each have a ticket id to link back to their parent ticket. Tickets themselves are all user generated so they contain errors and inaccuracies.

Linking computers to their tickets is not as straightforward as following an id. Each ticket is linked via glpi_items_tickets. This table contains the id of the ticket and the id of an item. What the item is, is determined by an extra row item type.

Centreon



Centreon is a hybrid IT monitoring system, historically it was called Oreon. It is based on the Nagios. The main service and software of Centreon is opensource but modules and extensions are provided for an extra cost, some of these can be substituted with open source variants developed by the community. It creates a simplified overview to heck on the status of systems. Status of systems is gathered using an agent that sends monitoring data using the SNMP protocol. On the Centreon server there is activated a trap that received the packets and monitors the values for each machine.

This is by far the largest database system. It has a size of 12GB and the monitoring data table we are interested in contains 178 515 000 rows. This large size complicates making changes to the database, so it is best to leave the large tables as is.

Information about the computers and systems is kept in the Host table. Each system is assigned a name and id these are then used to identify the systems in the rest of monitoring system. Monitor data is all stored in a table called data_bin. To data_bin has a value, time, and metrics_id, this metrics_id id linked to a row in the metric table which describes which value is monitored. The link between computers and metrics is mate using the index_data table this table links a host_id to a metric_id.

Combination of databases

Now that we have an idea what we can find where in the database it is possible to think about linking the computers in each table. Because there is no clear link between each database, I will compare the different available unique fields for each computer and investigate the possibility to make a link based on these parameters.

Representation of computers

OCS: Device ID: TODA300070855-2014-04-08-14-13-48
Name: TODA300070855
UUID: 585E6580-682B-11E2-9400-10604B7DF1AD

GLPI: Name: TODA300070855
Serial: CZC3061PS5
Other Serial: A300070855
UUID: 585E6580-682B-11E2-9400-10604B7DF1AD

Centreon: host_name: T24_TODA300070855
host_alias: TODA300070855

The only field that can be used to link all 3 database systems is the name field in glpi and ocs and the host_alias in Centreon. This is not perfect, these 3 database systems contain a different number of computers in these tables, many also contain duplicates in all rows. For our application it is not important to get all data at this moment the importance is linking.

We link using a center join so if there are no matches, we ignore the devices. If we go through with selecting based on name, we end with 206 devices. These are each given a new unique id in the combined database and combined in a row with the device name and ids of the computer in the 3 databases.

OCSWeb

To construct this part in the combined database it is enough to just get all computers from hardware that are in the combined table. Afterwards we add another field to the hardware table with the new id from the combined database to make it easy to refer to a computer. For the software table in ocs the same steps are taken. Afterwards we end up with 2 tables we can easily reference from the combined database.

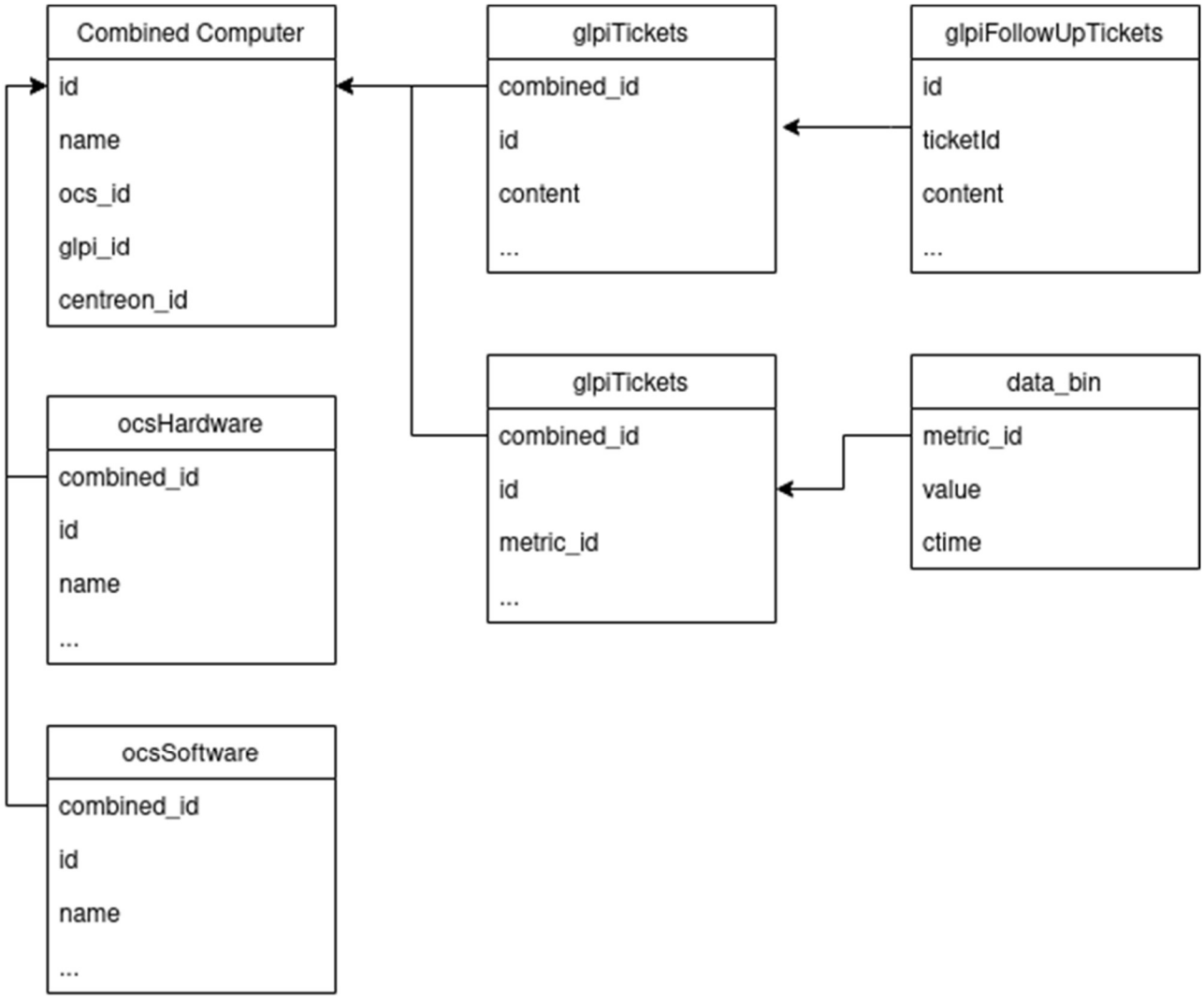
GLPI

Here we go through the computer table and the link these directly with a ticket by adding the id of the combined database directly to the ticket-list and add this new list to the combined database. In this way we filer out all tickets that are not linked to computers. We also filter the followup-tickets to only include the relevant tickets and add this table to the combined database.

Centreon

In the centreon table we run through the hosts select the relevant rows and combine index_data and metrics. We end up with one table which can be referenced with the id from the combined computer table. The actual metric data we keep in the Centreon database for now because of the large size moving this is impractical.

Overview



After all these steps we end up with following database design for our combined data. This will be a good basis for our API and makes it clear how to navigate through all available information about each machine.

Future

At this moment, the integration and selection of data is done manually. This is not a future proof method because it is not certain that the database structure will remain the same if software is upgraded. Another problem is we need full read access to all databases. It is also not officially supported by any of the software packages. So, I would propose to research the ability to write plugins for each of these and make them log all relevant data to the combined database automatically. Following I will give a brief overview of possible available options, these will need more research and testing to determine if they are viable for use in a future version of alertia.

Glpi

Has some support for exporting data. WebServices runs a server along side glpi that serves as an API endpoint that external applications can use to check and control glpi. There is also dumpentity to routinely generate csv files and deploy these to the alertia server. The final option would be to write a custom plugin.

OcsWeb

Here there is the ability to setup a REST api server which can be used to access the information in the database and manage some functionalities.

Centreon

The best way to get the data out the centreon databases is configuring a connection to a timeseries database. This is a powerful tool and gives a lot of options to organize and categorize the data. From a developer perspective switching to using timeseries databases is worthwhile investment because it will significantly speed up access to metrics data.