Ariadne: a system for evaluating AMAZED's efficiency

R. C. Borges, S. Arnouts, P-Y. Chabaud, F. Fauchier, M. Gray, S. Jamal, V. Le Brun, O. Le Fèvre, A. Schmitt, C. Surace, D. Vibert & C. Vidal
Aix Marseille Univ, CNRS, LAM, Laboratoire d'Astrophysique de Marseille, Marseille, France

Abstract

Here we present Ariadne, an automation subsystem for AMAZED, whose primary task is to compute AMAZED's redshift estimation efficiency. Secondarily, it serves as structured repository of processed reductions and metadata, allowing users to find existing results specifying abstract criteria (such as astronomical object parameter intervals), and remotely execute software to generate results not yet in the repository.

Introduction

LAM (Laboratoire d'Astrophysique de Marseille) is at the forefront of development of next generation spectrographic instruments, such as PFS (Prime Focus Spectrograph - see Tamura 2016) and Euclid (see Laurejis 2014). For both projects, LAM is responsible for delivering a pipeline that estimates redshift values.
AMAZED (Algorithms for Massive Automatic Z Evaluation and Determination) is a project developed at LAM to deliver pipelines for PFS and Euclid (see Schmitt 2017).
AMAZED's efficiency is defined for spectra which have reference redshift values.

$$Z_{Error} = |Z_{AMAZED} - Z_{Reference}|/(1 + Z_{Reference})$$
$$Efficiency_{PFS} = Count(Z_{Error} < 10^{-4})$$

Efficiency depends on AMAZED's version and configuration; also on the reference spectra. These conditions change often during development, and will change during survey operations.
Ariadne is LAM's project to automate usage of AMAZED.

Objectives

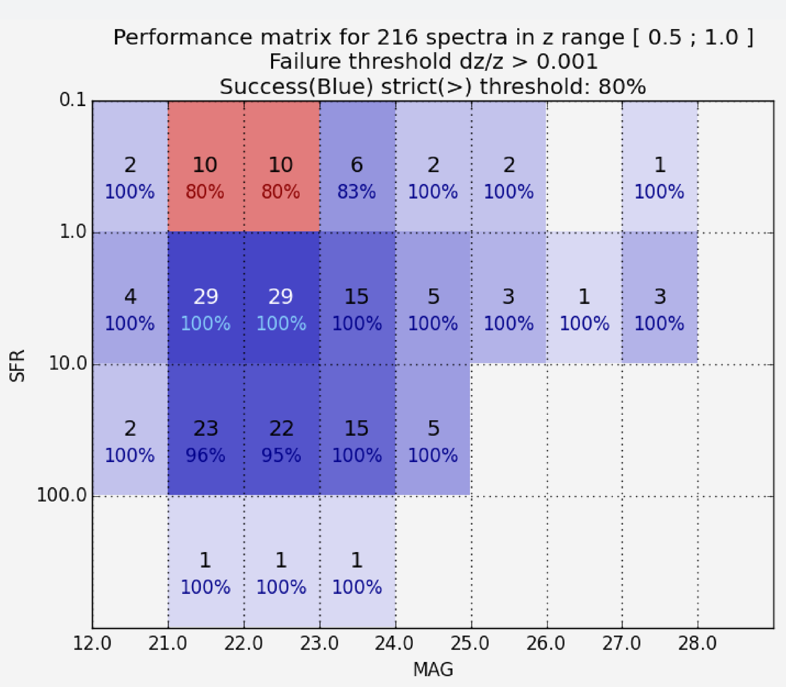

Figure: AMAZED efficiency table.

The principal objective is to produce "efficiency tables" such as the one displayed on figure 2.
Each table is linked to a specific version and configuration of AMAZED, and to a set of reference spectra.
These tables illustrate the performance of AMAZED with regards to intervals of its input parameters. This allows to verify if AMAZED performs within its intended precision, and suggests the parameter intervals where the pipeline is failing.

Since each table depends on several factors, assessing AMAZED's global efficiency suffers from an excessive number of test cases. Ariadne handles the repetitive tasks and does the bookeeping, so users only deal with efficiency in an abstract manner.

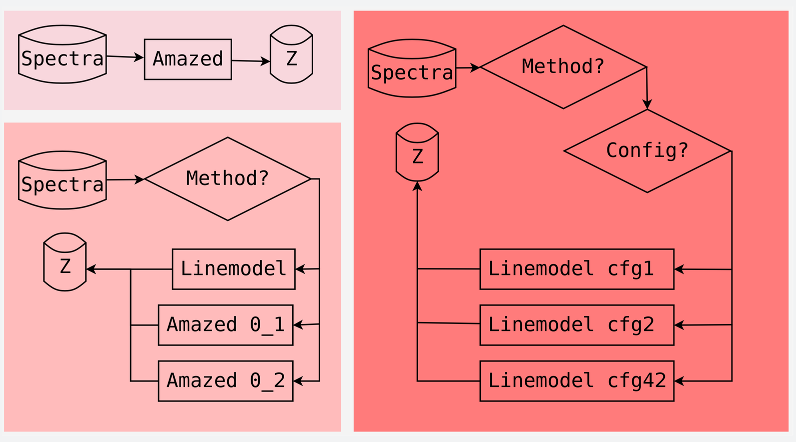

Figure: The explosive growth of AMAZED cases.

Secondarily, Ariadne was built modularly, and its components can be reused to mock telescope operations, and in this fashion we can test AMAZED's performance under conditions closer to its intended usage.

Technologies

Ariadne is developed through Agile practices and designed to be highly modular. It is written in Python 3. Some of Ariadne's modules use specific packages:

- **Pépin** is based on ZeroMQ;
- **Minos database** is based on SQLAlchemy (see Figure 4);
- **Ariadne** uses PyQt.

Development is done in a GitLab instance at LAM, with continuous integration with our Jenkins instance. Unit tests use unittest and nose.

Architecture

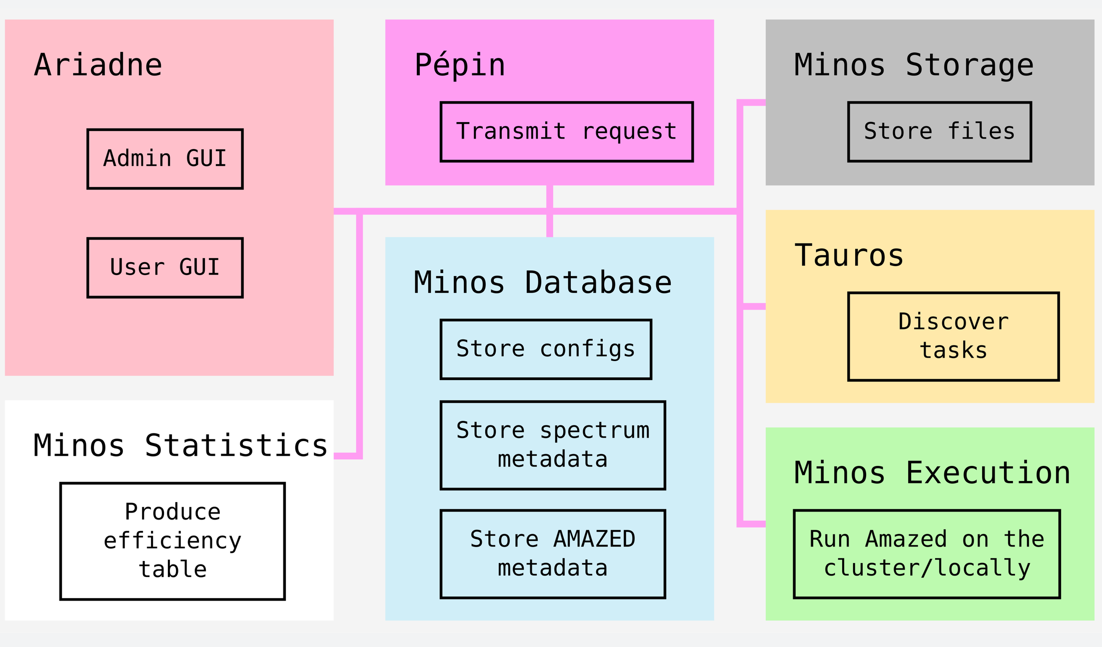

Figure: The modules that comprise Ariadne.

Ariadne's architecture is that of a set of daemons for the backend system. The frontend have Admin and User's GUIs which are multithreaded applications. We will deploy also a web client.

Design

Ariadne's database stores metadata about AMAZED, ProcAOS and the samples they process. Automation is built as logic around the database.

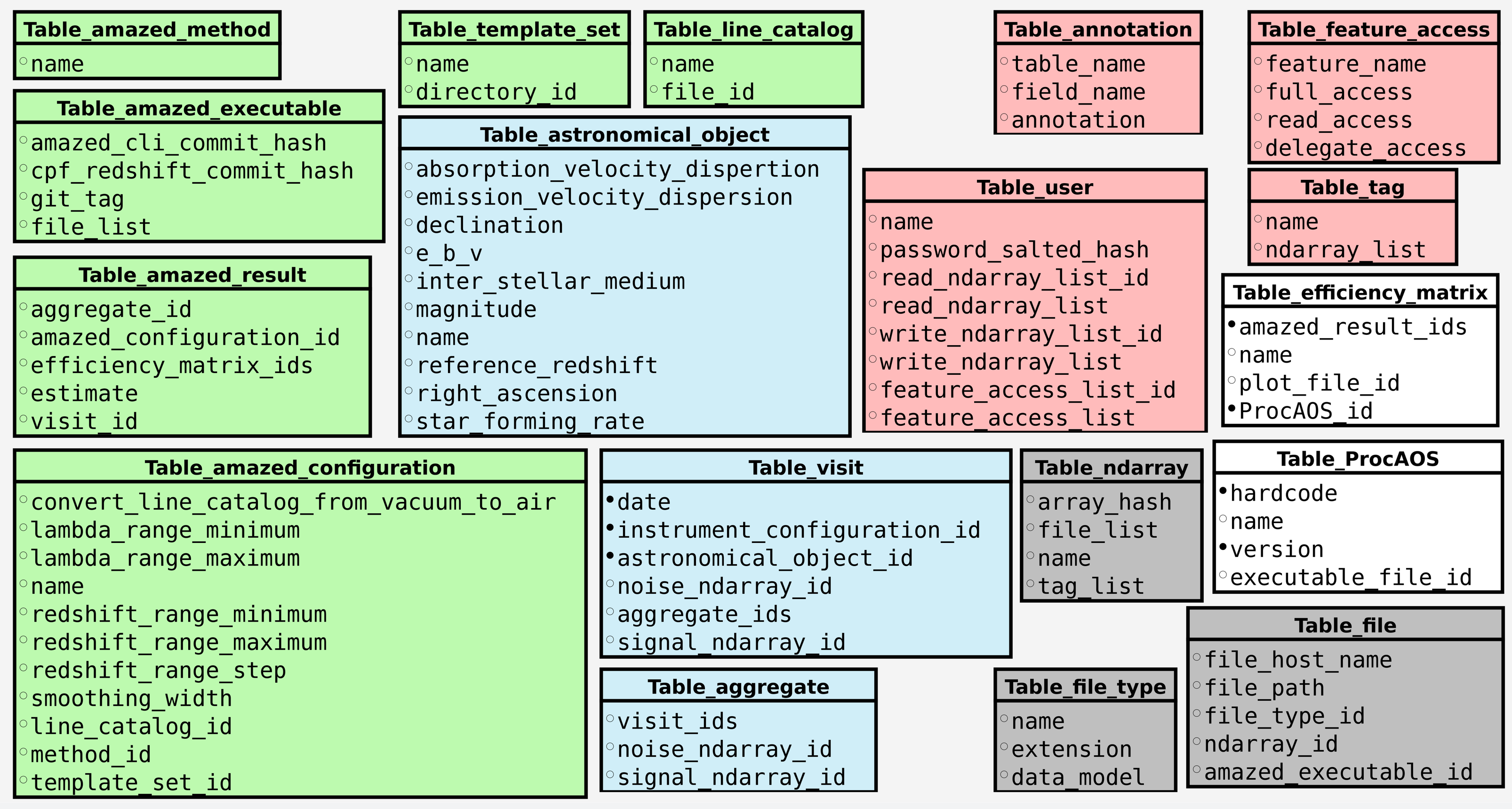

Figure: 4 - Ariadne's database tables.

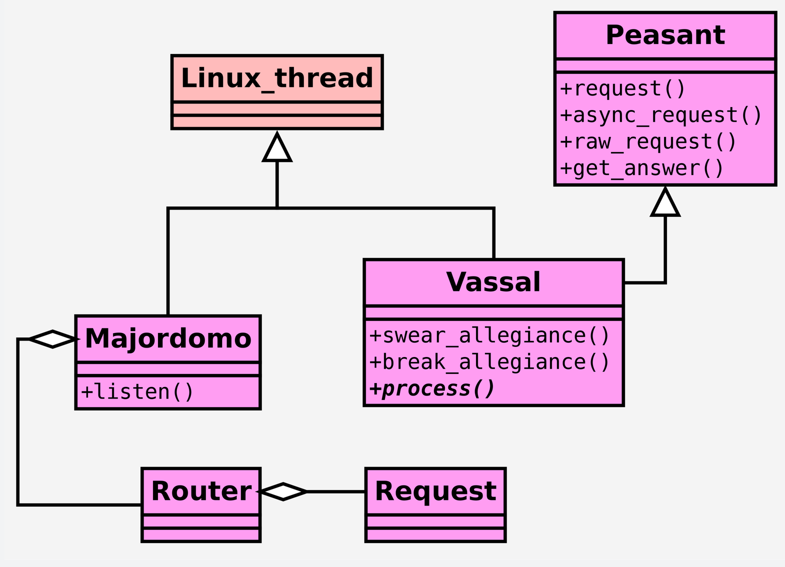

Figure: UML diagram of the classes in *Pepin.API*.

Pépin is built on top of ZeroMQ's Majordomo pattern. It has an API subpackage with Majordomo, Vassal and Peasant classes. Vassals provide services, while Peasants consume them. Daemons inherit from Linux_process, while volatile processes are Linux_threads. As long as clients send well-formed requests, they can use the services.

Conclusions

Ariadne currently allows a user to abstract many details of the complex task of producing AMAZED's efficiency tables. This is a crucial element for allowing developers to conveniently assess impacts of new code in AMAZED.
The modular design of Ariadne amde easy to be refactor Pépin into an independent project.
The administrative GUI allows to manage a large data collection with automated tools.

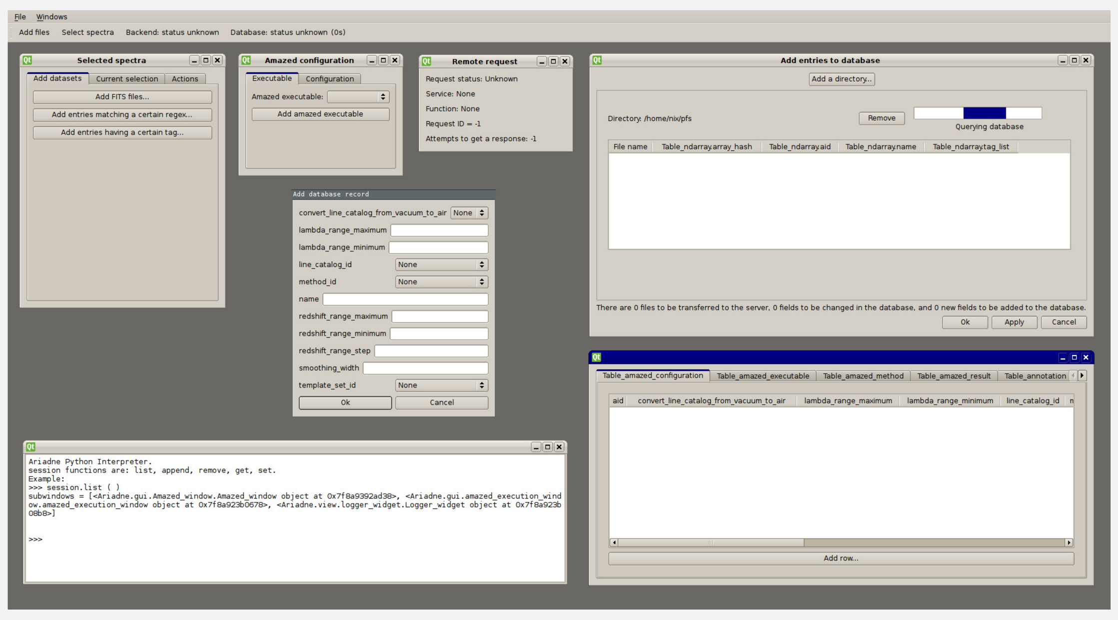

Figure: 6 - Ariadne's administrative GUI.

Future research

The two projects we intend to attack next are Tauros and WebZ, the first being an automation module for Ariadne that will be responsible for discovering tasks that should be executed in order to produce up-to-date efficiency tables, given a template.
The WebZ project is a user-friendly interface to Ariadne's services, which will eliminate the need for installing client GUI software on a user's machine.

References

Schmitt, A. et al, AMAZED: Algorithm for Massive Automated Z Evaluation and Determination, ASP Conf. Ser., 2017.
Tamura, N. et al, Prime Focus Spectrograph (PFS) for the Subaru telescope: overview, recent progress, and future perspectives, Proc. SPIE 9908, 2016.
Laureijs, R. et al, The Euclid Mission: Cosmology Data Processing and Much More, ASP Conf. Ser., vol. 485, 2014.

Acknowledgements

This project would not have been possible without grants from CNRS, connected to the PFS international collaboration.