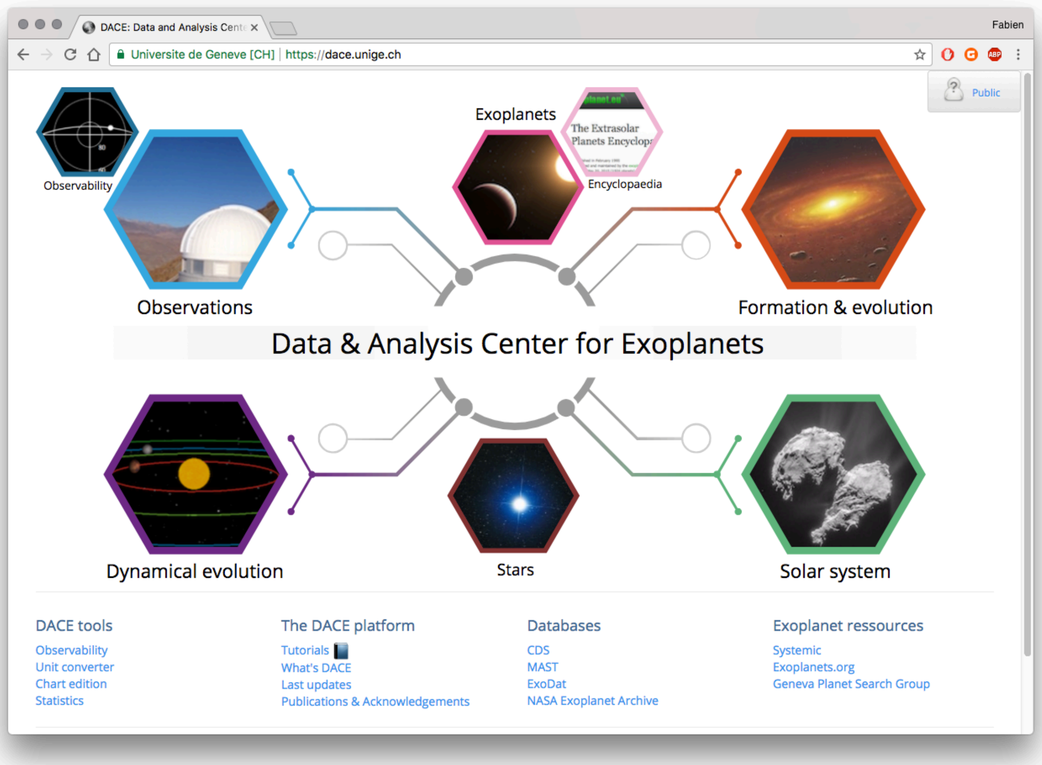
DACE: Development, tests and deployments of web applications

J. Burnier, F. Alesina, N. Buchschacher and D. Ségransan



DACE: Data Analysis Center for Exoplanets

J. Burnier

The Data and Analysis Center for Exoplanets (DACE) is a web platform based at the University of Geneva (CH) dedicated to extrasolar planets data visualisation, exchange and analysis.
This platform is based on web technologies using common programming languages like HTML and Javascript for the front-end and a Java REST API for the back-end.

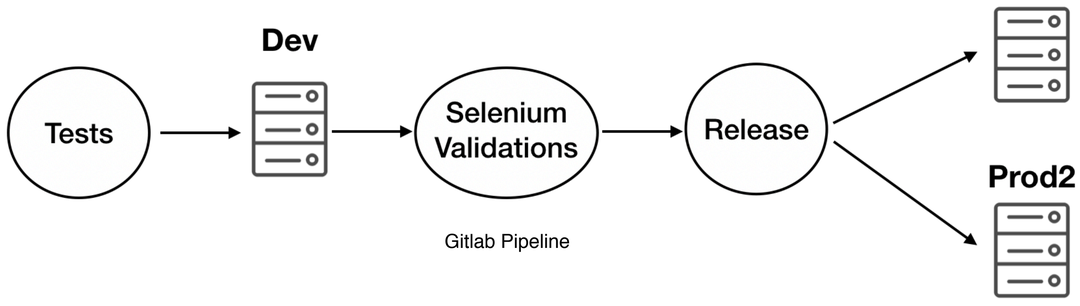
Over the last 12 months, the process to maintain, develop, test and deploy the applications has been dramatically improved to facilitate the maintenance and the integration of new features. The goal of such automation is to let more time to focus on development and reduce the duplicated work.
To achieve this result, we migrated our Java application to the Maven software project management. We implemented a pipeline on GitLab which consists of executing the tests and deploy the application in a dev environment at every commit. We added Selenium tests to simulate the user and compare the new commits with the old ones.

Maven and unit tests

The first thing to do was to encapsulate existing code to a project management tool which manage dependencies and simplify compilation, test and jar/war creation. We chose maven as it remains the reference on Java. Then we added unit test to existing code where possible. To do this we followed mvn convention and started using JUnit. Finally, the idea for backend API for example, was to apply Test Driven Development when possible.

Gitlab: continuous integration tool

We needed a tool to do continuous integration. Fortunately, other project inside our university started to use Gitlab. We migrated our project from svn to git and pushed the code inside gitlab. After that, a gitlab-runner must be installed where you want to run your build. And finally, we added a *.gitlab-ci* file into every project. You can see below the pipeline we created on DACE



Gitlab Pipeline

Selenium: website test tool

Before going to production, web tests should be done to ensure having no bugs and no regression (same as unit test but on visual part). To do this we use selenium. This tool simulates user interaction on a website. To facilitate selenium test development, we adopted Page Object Pattern.

Example of home page test :

```java
package ch.unige.dace.pages.home;

import ...

/**
 * Created by julien on 20.10.17.
 * <p>
 * Test the home page. Use FunctionalTest to setup selenium environment
 */
public class HomePagePolygonsTest extends FunctionalTest {

    @Test
    public void shouldDisplayHomePage() throws InterruptedException {

        Navigator.openDaceWebsite();

        HomePagePolygons homePagePolygons = new HomePagePolygons();

        assertThat(homePagePolygons.isDisplayed()).isTrue();

        List<LocatorAndPage> polygons = homePagePolygons.getPolygons();

        for (LocatorAndPage polygon : polygons) {
            Page page = Navigator.goToPage(polygon);
            if (Navigator.isPageOpenOnNewTab()) {
                Navigator.switchTab();
                assertThat(page.isDisplayed()).isTrue();
                Navigator.closeTab();
            } else {
                assertThat(page.isDisplayed()).isTrue();
                Navigator.goBack();
            }
        }
    }
}
```

Backend and Frontend deployment

The deployment is also handled by gitlab. It needs to be defined in *.gitlab-ci.yml*. On our side, after every commit, we deploy backend and frontend on dev environment.

Then if everything is ok, the app is released, deployed manually on one production server and after some manual tests, deployed on the other production server.

You can see below an example of *gitlab-ci.yml* to build and deploy on staging

```yaml
stages:
  - build
  - staging
  - release
  - production

Maven Test and Build:
  stage: build
  script: mvn clean install
  # Need this config to pass the jar to next stages
  artifacts:
      paths:
        - target/*.war
      expire_in: 1 day

Deploy to Staging:
  stage: staging
  environment:
    name: Staging
  script:
    - export WEBAPP_FULL_NAME=$(basename target/*.war)
    - cp target/$WEBAPP_FULL_NAME /opt/tomcat/dace-webapps
    # ln -fsn will do the following : -f overrides if symlink already exists. -n will not follow previous symlink.
    # -s : standard option to have a symlink and not a hard link
    - export WEBAPP_NAME=$(echo $WEBAPP_FULL_NAME | sed 's/-.*//')
    - export WEBAPP_NAME="${WEBAPP_NAME}.war"
    - ln -fsn /opt/tomcat/dace-webapps/$WEBAPP_FULL_NAME /opt/tomcat/webapps/$WEBAPP_NAME
```

https://dace.unige.ch

Swiss National Science Foundation

The National Centres of Competence in Research (NCCR) are a research instrument of the Swiss National Science Foundation