Final Exercise

DevOps Dataset

Guillermo Vigueras and Ángel Herranz
{gvigueras,aherranz}@fi.upm.es

Programming for Data Processing
DLSIIS-UPM

April 21, 2020

1 Introduction

Your group is working within the Data Analysis Dept. of a company related with software development. The company has an important DevOps team whose manager has requested the Data Analysis Dept. to give some insight based on the data generated by developers.

2 SCRUM Methodology

The DevOps team uses the SCRUM methodology for organizing the development tasks (see Figure 1). The different development requirements are collected within the so called *Backlog*.

The DevOps team uses the following elements within the BackLog:

- **Feature**: represents an application functionality from the business point of view.
- **User Story**: represents a specific functionality of finer grain derived from a Feature, which must be developed within the framework of a sprint.
- **Bug**: Any errors or failures reported in the application.
- **Work Item**: Work item to represent any other situation that may occur.

Regarding the elements that represent periods of time, the main one is the *Sprint*. A sprint represents a small period of time (in practice the DevOps team

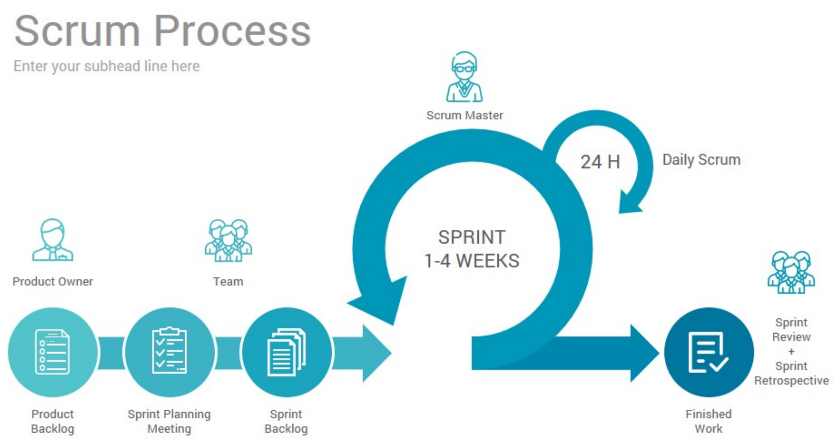

Figure 1: SCRUM process

defines sprints of one week) in which the developers must accomplish a series of elements from the Backlog.

Concretely, the sprints for which the data analysis is requested to be performed, involve the following dates and times:

- **Sprint 1** - begin: “2019-09-30 00:00:00”, end: “2019-10-06 23:59:59”
- **Sprint 2** - begin: “2019-10-07 00:00:00”, end: “2019-10-13 23:59:59”
- **Sprint 3** - begin: “2019-10-14 00:00:00”, end: “2019-10-20 23:59:59”,
- **Sprint 4** - begin: “2019-10-21 00:00:00”, end: “2019-10-27 23:59:59”,
- **Sprint 5** - begin: “2019-10-28 00:00:00”, end: “2019-11-03 23:59:59”

On the other hand, SCRUM defines three roles, the Product Owner, Scrum Master and Development Team Member.

- **Product owner**: It is the one that ensures that the team works properly from the business perspective. It has among its functions to help define the User Story, prioritize them and include them in the backlog. Therefore, the Product owner is expected to be the user who creates both the features and the User story. He will also be in charge of creating the meetings, assigning them to himself and closing them.
- **ScrumMaster**: Facilitator, ensures the rules are applied. He/she is in charge of assigning the User Storys to the different Sprint and users, and will be in charge of activities such as reassignment, reopening and replanning of Issues, as well as the creation of a bug.

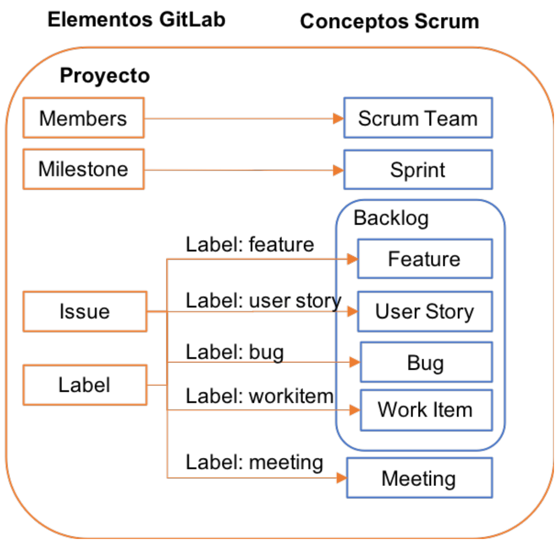

Figure 2: Mapping of SCRUM to GitLab concepts

- **Development team member**: This role is in charge of delivering the product. This role performs the code commits and close the issues. In practice, everyone fulfills this role, even if they also have the role of Product owner or ScrumMaster.

3 Record of SCRUM data

The DevOps team uses GitLab as a framework for collaborative development. In order to register SCRUM data of developers, a mapping of SCRUM to GitLab concepts has been defined. Figure 2 shows such mapping.

Thus, a Sprint will be named in GitLab as a *Milestone*. The same for Backlog elements like: Feature, User Story, Bug and Work Item. They are represented through a GitLab *Issue*, plus a GitLab *Label* which transforms the Issue into a SCRUM element. It is important to note, that this mapping of SCRUM to GitLab concepts is just a naming aspect, and something to be taking into account when processing the data files containing development information.

On the other hand the DevOps team uses GitLab as a control version framework for developing collaboratively. Figure 3 shows the main Git flow for synchronising each local copy to the remote repository hosted in the server. The main information regarding Git flow to be taken into account by Data Analysts is the fact that developers perform *commits* for submitting their changes to the code. This will be the basic information for providing insight regarding developers activity. See Section 4.4 for more information registered for each commit.

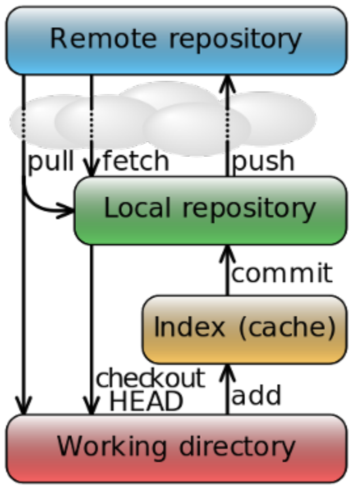

Figure 3: Caption

4 Description of Dataset

This section describes the files containing the DevOps recorded information. The information is provided in comma separated value (.csv) files, where the separator is the character `'|'`.

Files described in this section are inter-related through columns containing the same type of data. These common columns across different files can be identified because they have the same name. As an example, files `usersByGroup.csv` and `commitsByGroupByProject.csv` are related through column `idGroup` (see Sections 4.1 and 4.4).

4.1 File usersByGroup.csv

This file contains data about the developement groups and users per group. There is one row per user.

- `idGroup`: id of the group
- `fullNameGroup`: name of the group
- `idUser`: id of the user
- `userName`: username of the user
- `rol`: SCRUM rol of the user. Where rol names used are:
  - `PO`: for Product Owner

  - `SM`: for SCRUM Master
  - `D`: for Developer

- `email`: email address of the user

4.2 File milestonesbyproject.csv

This file contains data about the SCRUM the issues per sprints by project (i.e. meeting, feature, user story, bug and work item). There is one row per issue defined by each developer group, linking each issue to a sprint as defined by each development group.

- `idGroup`: id of the group
- `idProyecto`: id of the project
- `idMilestone`: id of the sprint
- `descMilestone`: description of the sprint
- `titleMilestone`: title of the sprint
- `createAtMilestone`: date and time of creation of the sprint
- `updateAtMilestone`: date and time of update of the sprint
- `startDateMilestone`: date of beginning for the sprint assigned by the development group
- `dueDateMilestone`: end date for the sprint assigned by the development group
- `stateMilestone`: state of the sprint (i.e. `closed` or `active`)
- `idIssue`: id of the issue
- `titleIssue`: title of the issue
- `stateIssue`: state of the issue (i.e. `closed` or not). In case the issue is closed, this column contains the string `closed`. In case the issue is still open, this column is empty.

4.3 File issuesbyproject.csv

This file contains data about SCRUM issues by project (i.e. meeting, feature, user story, bug and work item). There is one row per issue.

- `idGroup`: id of the group
- `idProject`: id of the project

- `idIssue`: id of the issue
- `descIssue`: description of the issue
- `titleIssue`: title of the issue
- `createAtIssue`: creation date and time of the issue
- `labelsIssue`: labels associated to the issue. The labels record the type of SCRUM item associated to this issue. Although many labels are allowed, only one of the following labels should appear:
  - `user-story`
  - `feature`
  - `meeting`
  - `bug`
  - `workitem`
- `UpdateAtIssue`: update date and time of the issue
- `stateIssue`: state of the issue (i.e. `closed` or not)
- `idAuthorIssue`: id of the user that created the issued
- `idAssigneeIssue`:

4.4 File commitsByGroupByProject.csv

This file contains data about commits by group and project. There is one row for each file within the commits. For example, if the total number of commits is 6 and each commit refers to 3 files, the total number of rows of `commitsByGroupByProject.csv` is: $6 * 3 = 18$ rows.

- `idGroup`: id of the group
- `idProject`: id of the project
- `commitId`: id of the commit
- `commitDate`: date of the commit
- `commitAuthor`: author of the commit
- `commitEmail`: email of the commit author
- `commitMessage`: message associated to commit
- `diffFileId`: id of the file within the commit. When multiple files are uploaded within the same commit, this field allows to differentiate the files by assigning and id.

- `diffFileName`: name of file within the commit
- `diffBytes`: size in bytes of the incremental change for file identified by `diffFileId`, within the commit.
- `diffLines`: number of lines of the incremental change for file identified by `diffFileId`, within the commit.

5 Data Analysis tasks

The manager of the DevOps team has many developers under its supervision, so she is not sure about team performance and compliance with SCRUM methodology.

According to the description of the data registered for developers, the DevOps team manager is asking the Data Analysis Dept. to give some insight and answer the following questions:

1. Compute a Pandas `DataFrame` for all groups showing the total number of commits for each group and the average commit size (in Bytes) per group.

   As an example, let's suppose a group with `fullNameGroup=19_20_grupo1` has 5 commits with sizes in bytes: 10, 5, 20, 5, and 10, respectively. Thus, total number of commits performed by the group would be 5 with an average commit size of 10 Bytes.

   The following Listing shows an example of the `DataFrame` returned. Note that although this example is referring just to group `19_20_grupo1`, the `DataFrame` must contain information for all groups.

```
  fullNameGroup   numCommits   avgCommitSize
0 19_20_grupo1    5            10.0
1 19_20_grupo2    ...          ...
2 19_20_grupo3    ...          ...
3 19_20_grupo4    ...          ...
  ...             ...          ...
```

2. Compute a Pandas `DataFrame` for all groups showing the percentage of closed and open issues of each group, with respect to the total number of issues registered by each group. The percentage must be expressed using a float number in the range [0-1].

   As an example, let's suppose group with `fullNameGroup=19_20_grupo1` has a total of number of issues of 20. Within this number of issues, 15 are closed and 5 are still open. So, in percentage, 75% are closed and 25% are still open.

   Example of the `DataFrame` returned. Note that although this example is referring just to group `19_20_grupo1`, the `DataFrame` must contain information for all groups.

```
  fullNameGroup   numIssues   closed  open
0 19_20_grupo1    20          75.0    25.0
1 19_20_grupo2    ...         ...     ...
2 19_20_grupo3    ...         ...     ...
3 19_20_grupo4    ...         ...     ...
  ...             ...         ...     ...
```

6 Submission Details

- Each group must submit the solution to the exercise using a python template file that can be found in Moodle.
- Data files, described in this document, can also be found in Moodle.
- Submission will be by email to: **gvigueras@fi.upm.es**.
- **Due date is 06/05/2020**.
- The template file must be renamed using the following convention:
  - `final_exercise_group-<groupNumber>.py`
- Subject of the email must be `[PDP-Group-<groupNumber>] Final exercise submission`.
- Example: for group 6, submission should be like:
  - Template file should be renamed to: `final_exercise_group-06.py`
  - Email subject should be like: `[PDP-Group-06] Final exercise submission`
- Note the 2 digits used for number 6.