DELIVERABLE 1: MISURARE LA STABILITÀ DI UN ATTRIBUTO DI PROGETTO

Martina De Maio

Studentessa Laurea Magistrale in Ingegneria Informatica,

Università degli Studi di Roma “Tor Vergata”, Roma, Italia

Matricola: 0296447

Agenda

La seguente presentazione ha lo scopo di fornire una descrizione del processo di analisi e studio effettuato per il Deliverable 1, che consiste nell'utilizzare uno strumento statistico per misurare la stabilità di un attributo di progetto.

Il lavoro è organizzato secondo la seguente scaletta:

1. Introduzione
2. Metodologia
3. Risultati
4. Links

Introduzione

- Valutare e misurare la stabilità di un attributo di progetto permette di monitorare, controllare e predirne il comportamento, con conseguente miglioramento della qualità del software sviluppato e ottimizzazione di tutto il processo di sviluppo.
- Il progetto analizzato è il progetto open source ***Apache Daffodil***, per il quale l'attributo da monitorare è il ***numero di bug risolti*** nel corso del tempo.
- Per far ciò, è necessario utilizzare uno strumento statistico, nel nostro caso il "***Process Control Chart*** ", in grado di mantenere sotto controllo i vari parametri di un processo.
- Dall'analisi del Process Control Chart è possibile determinare la stabilità del processo in esame rispetto all'attributo di interesse, in questo caso il numero di bug risolti al mese, osservando l'andamento di tale attributo rispetto a dei valori di soglia opportunamente calcolati.

Introduzione

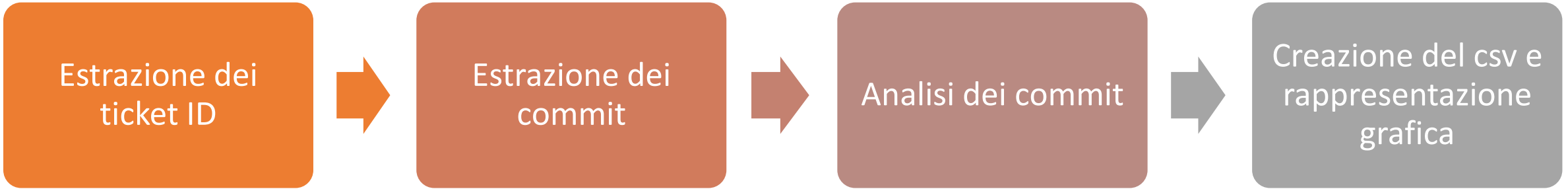
- Il periodo analizzato è di circa 10 anni, compreso tra Gennaio 2012 e Dicembre 2021.
- La rappresentazione grafica ricavata mette in relazione il numero di bug risolti (indicati sull'asse Y) al mese (asse X), con delle soglie precalcolate, allo scopo di analizzarne l'andamento e la stabilità.
- Dall'analisi del grafico si valuta la qualità del processo di sviluppo, studiando eventuali comportamenti anomali rispetto alle soglie e analizzandone le motivazioni, allo scopo di eliminarne le cause e quindi raggiungere la stabilità richiesta.
- Gli strumenti utilizzati per la realizzazione del Deliverable 1 sono:
  - Il sistema di ***Ticket Tracking*** offerto da ***JIRA***
  - Il sistema di ***Repository*** offerto da ***GitHub***
  - ***SonarCloud***, servizio di misurazione della qualità e sicurezza del codice, cloud-based.

Metodologia

I passi seguiti per raccogliere i dati e ricavare la rappresentazione grafica dell'andamento dell'attributo di interesse sono stati i seguenti:

1. Estrazione degli ID dei ticket in JIRA
2. Ricerca ed estrazione, tramite la repository GitHub, di tutti i commit del progetto
3. Analisi dei commit ed estrazione di quelli il cui messaggio contiene gli ID dei ticket estratti al punto 1
4. Creazione di un file .csv da cui ricavare la rappresentazione grafica del Process Control Chart, che mostra l'andamento dei difetti risolti nel progetto in questione.

Per effettuare la raccolta dei dati è stato necessario utilizzare le piattaforme di ***Jira*** e di ***GitHub***, facendo uso rispettivamente dell'API di ***Json*** e di ***JGit***.



Metodologia: Ricerca di fixed tickets in JIRA

Il sistema ***Ticket Tracking*** offerto da JIRA ha permesso di raccogliere, tramite una query, gli ID dei ticket aventi:

- issueType = “ *Bug*”
- status = “*closed*” o ”*resolved*”
- resolution = “ *fixed* ”

```
j = i + 1000;
String url = "https://issues.apache.org/jira/rest/api/2/search?jql=project=%22"
        + projName + "%22AND%22issueType%22=%22Bug%22AND(%22status%22=%22closed%22OR"
        + "%22status%22=%22resolved%22)AND%22resolution%22=%22fixed%22&fields=key,resolutiondate,versions,created&startAt="
        + i.toString() + "&maxResults=" + j.toString();
try
{
JSONObject json = readJsonFromUrl(url);
```

Questa query restituisce un file *.json*, che può essere analizzato per ottenere le informazioni volute: in questo caso la chiave, ossia l’ID, dei bug.

Metodologia: Ricerca di commit contenenti ID dei ticket da GitHub

- Una volta effettuato il clone della repository da GitHub, si ottiene il log di tutti i commit del progetto, i quali vengono salvati in una lista di oggetti *RevCommit*.
- Tale lista viene poi analizzata e vengono associati, ad ogni ticket ricavato precedentemente, una lista di commit contenente nel messaggio l'ID del ticket.
- Tramite i commit si possono ricavare le date di risoluzione di ogni ticket, le quali corrispondono alla data dell'ultimo commit relativo ad ogni ticket.
- Bisogna far notare che non tutti i ticket presi da JIRA hanno un ID presente nei messaggi dei vari commit, mentre ci sono ticket che possono avere più di un commit associato. Nel primo caso, i ticket in questione vengono semplicemente scartati, mentre nel secondo caso si prende la data più recente e la si definisce *resolution date* per questi ticket.

Metodologia: Ricerca di commit contenenti ID dei ticket da GitHub

- L'utilizzo di GitHub per trovare la data di risoluzione di un ticket si è reso necessario perché in JIRA questa non sempre è fornita, o, se disponibile, spesso è da ritenere non affidabile (per esempio inferiore alla data di creazione del ticket).
- Ciò è dovuto al fatto che JIRA, non essendo completamente automatizzato, potrebbe riportare delle informazioni errate, scenario non possibile su GitHub, in quanto la data in cui avviene un commit viene impostata direttamente dal sistema e non è modificabile manualmente.

Metodologia: Ricerca di commit contenenti ID dei ticket da GitHub

- La ricerca degli ID nel log dei commit è stata laboriosa, poiché da una pre-analisi dei risultati si è verificato un andamento anomalo nel primo periodo di osservazione, in cui i commit non avevano alcun ticket associato, quindi non contenevano nessun ticket ID nel loro messaggio.
- Da un'analisi più approfondita è emerso che in tali commit iniziali l'ID riportato all'interno del messaggio non era in formato «DAFFODIL-*», che è quello ricavato da JIRA, ma in formato «DFDL-*», che quindi non veniva mai trovato. Si è estesa quindi la ricerca anche a tale stringa, ottenendo dei dati significativi su tutto il periodo di osservazione.
- Da un'ulteriore analisi dei risultati, si è notato che nel processo di ricerca delle stringhe, in alcuni casi queste venivano conteggiate come sottostringa di stringhe più lunghe. Per fare un esempio, nel caso di un ticket con ID «DAFFODIL-41», venivano erroneamente associati ad esso anche i commit contenenti la stringa, ad esempio, «DAFFODIL-415», fornendo risultati finali alterati. Tale problema è stato risolto affinando il processo di ricerca, estendendolo a tutti i possibili caratteri successivi, che, analizzando tutto il log del progetto, sono risultati essere i seguenti:

```
(commit.contains(ticketID1 +",") || commit.contains(ticketID1 +"\r") || commit.contains(ticketID1 +"\n")|| commit.contains(ticketID1 + " ") || commit.contains(ticketID1 +":")
          || commit.contains(ticketID1 +".")|| commit.contains(ticketID1 +")")|| commit.contains(ticketID2 +")") || commit.contains(ticketID2 + ",") || commit.endsWith(ticketID1) || commit.endsWith(ticketID2)
          commit.contains(ticketID2 + "\r")|| commit.contains(ticketID2+"\n") || commit.contains(ticketID2 + " ") || commit.contains(ticketID2 + ":") || commit.contains(ticketID2 + ".")) {
```

Metodologia: Creazione del csv

Terminata la raccolta delle informazioni, si crea una *SortedMap* avente come *key* l'anno e come *value* una lista di mesi, la quale rappresenta le entry del csv. In particolare, i passi seguiti sono stati i seguenti:

1. Si scorre ogni ticket e si prende l'anno e il mese della relativa resolution date.
2. Se l'anno in questione non è presente nella *SortedMap*, si crea una nuova entry della mappa avente come chiave l'anno e come value una lista in cui si inserisce il mese relativo al punto 1.
3. Se invece è già presente una entry avente come chiave l'anno, si aggiunge, alla lista di mesi associata a tale entry, il mese in questione.
4. Essendo una *SortedMap*, gli anni (le chiavi) saranno inseriti nella mappa in ordine cronologico, dal meno recente al più recente. Si nota che la lista dei mesi associata ad ogni anno può anche avere un mese ripetuto più volte.
5. Una volta creata la *SortedMap* contenente le entry da scrivere nel file csv, si scorre tale mappa e, per ogni anno trovato, si contano le occorrenze di ogni mese all'interno della lista associata a tale key, e si riporta tutto all'interno del file *.csv*.

Metodologia: Creazione del csv

Una volta raccolti e analizzati, i dati ottenuti vengono salvati in formato tabellare all'interno di un file **.csv**, avente come prima colonna il mese-anno, e come seconda colonna il numero di bug risolti in quella specifica data, come mostrato nel seguente estratto.

Month-Year	Number of fixed Bugs
JANUARY - 2012	0
FEBRUARY - 2012	0
MARCH - 2012	0
APRIL - 2012	1
MAY - 2012	0
JUNE - 2012	1
JULY - 2012	2
AUGUST - 2012	8
SEPTEMBER - 2012	10
OCTOBER - 2012	12
NOVEMBER - 2012	13
DECEMBER - 2012	7
JANUARY - 2013	8
FEBRUARY - 2013	18
MARCH - 2013	19

Metodologia: Rappresentazione grafica

Oltre al file *.csv* è stato creato un ulteriore file in formato *Excel (.xlsx)*, nel quale sono state aggiunte 4 colonne necessarie per la realizzazione del Process Control Chart:

- Media
- Deviazione standard
- Upper limit
- Lower limit

Il ***lower limit*** e l'***upper limit*** servono a classificare la stabilità di un processo di sviluppo, in quanto rappresentano i limiti entro cui un attributo (ciò che si sta misurando, in questo caso il numero di bug fixati) sia considerato stabile: quando i valori cadono al di fuori dei due limiti, significa che in quell'intervallo temporale ci si trova in uno stato d'instabilità.

Per computare i due estremi sono state applicate le seguenti formule:

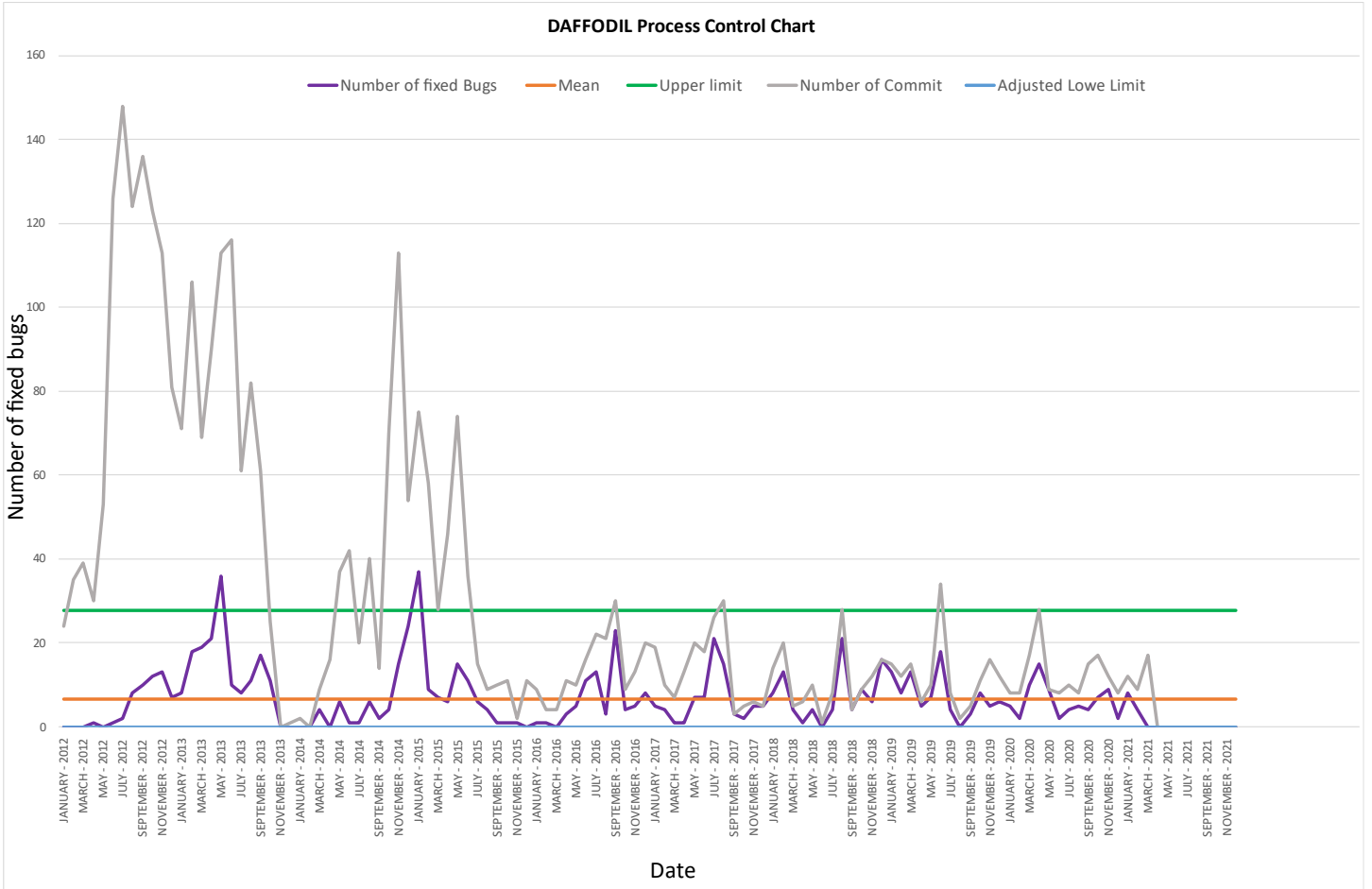
- $upper\ bound\ = media\ + 3 * \ deviazione\ standard$
- $lower\ bound\ = media - 3 * deviazione\ standard$

Risultati

Il Process Control Chart ottenuto, che mostra l'andamento dei difetti risolti nel progetto apache DAFFODIL negli ultimi 10 anni, è riportato di lato.

Sull'asse X sono riportati i mesi del periodo di osservazione, per ognuno dei quali, sull'asse Y, è riportato il corrispondente numero di bug risolti, indicati dalla curva di colore viola.

Sul grafico sono riportati anche i valori dell'***Upper Limit*** (in verde), dell'***Adjusted Lower Limit*** (in giallo), della ***Media*** (in arancione), e del numero di commit effettuati (in grigio).



Process Control Chart relativo al progetto DAFFODILS. Il Lower Limit è stato «aggiustato» e portato da un valore negativo a 0.

Risultati

- I tickets presi inizialmente da JIRA sono 1014, dei quali solo 829 hanno una corrispondenza nei 3503 commit effettuati sul progetto, dunque i dati ottenuti si riferiscono a circa l'82% dei tickets.
- Il lower limit ottenuto ha un valore di -14,57024 , per cui si è considerato ragionevole sostituire, nel Process Control Chart, a tale valore un «***Adjusted Lower Limit***» pari a 0, poiché, essendo l'unità di misura il numero di bug fixati in una determinata data, è impossibile che tale numero cada sotto lo 0.
- Durante le fasi iniziali si possono osservare oscillazioni più marcate, sintomi di un comportamento instabile del processo, con il superamento dell'upper limit in due casi: Maggio 2013 e Gennaio 2015. Questo comportamento è spiegabile dal fatto che in quel periodo vengono registrati un numero molto elevato di commit, come si può vedere nel grafico.

Risultati

- In una seconda fase, da Maggio 2016 a Luglio 2019, il processo di sviluppo presenta oscillazioni meno marcate, sintomo di un processo di maturamento dello sviluppo, seppur con valori ancora lontani dal valor medio.
- Con il tempo il processo di sviluppo si è ulteriormente stabilizzato, infatti da Luglio 2019 i valori si attestano tutti vicino alla media (linea arancione) senza grosse variazioni, tutti abbondantemente dentro i limiti superiori/inferiori.

Introduzione

- La repository contenente il codice sorgente del progetto Apache oggetto di analisi, di cui è stato fatto il clone, è disponibile al seguente link: https://github.com/apache/daffodil

- L'estrazione e la raccolta dei dati è stata realizzata tramite codice Java, disponibile alla seguente repository di GitHub: https://github.com/martina97/ISW2_21-Deliverable1

- Di seguito il link con i risultati dell'analisi sulla qualità del software effettuata con SonarCloud e lo screenshot riepilogativo: https://sonarcloud.io/dashboard?id=martina97_ISW2_21-Deliverable1