SVILUPPO SOFTWARE

Come già detto, non seguendo in maniera precisa una metodologia di sviluppo software, né agile né plan-based, la giornata lavorativa non è rigorosamente definita (TODO: da decidere se lasciare come frase o meno) e ogni membro del team può ritrovarsi a svolgere attività diverse all’interno dello stesso ciclo di sviluppo in relazione anche ai propri impegni legati ai progetti di ricerca.

L’assegnamento e la coordinazione di queste attività vengono supervisionati dal lead developer del progetto, il quale ne detiene la responsabilità e perciò si assicura, tramite un **costante rapporto diretto** tra i membri del team, che il lavoro venga svolto correttamente e efficientemente.

Ciò è ottenibile grazie all’utilizzo giornaliero di Slack, software di collaborazione aziendale utilizzato per scambiare messaggi e documenti in modo istantaneo all’interno del team. Esso (TODO: si intende il team), pur non effettuando regolari meeting informativi, ha a disposizione in ogni momento il documento dei requisiti (TODO: da capire se esiste davvero o meno) per la prossima release, costantemente aggiornato in modo da fornire una panoramica riguardante lo stato attuale/corrente del progetto. Nel caso uno dei membri si trovi in difficoltà riguardo una sezione particolarmente intricata del codice, egli ha modo di contattare immediatamente gli altri e, se necessario, viene organizzato un meeting al fine di sciogliere i dubbi e non incagliarsi con il proseguimento dello stesso (TODO: si intende del codice). In presenza di situazioni più spinose, può essere prevista una sessione di *pair programming*, durante la quale due membri del team lavorano contemporaneamente all’implementazione di una stessa parte di codice scambiandosi regolarmente il ruolo di “*driver*” (colui che scrive il codice e espone una possibile implementazione funzionante) e “*navigator*” (colui che lo controlla e propone soluzioni alternative) (TODO: volendo si può anche accorciare, ho scritto qualcosa in più per spiegare meglio ma magari non serve).

Nonostante si faccia largo uso di test all’interno del progetto, Tribuo non aderisce rigorosamente al modello del **test driven development**. Infatti, a seconda della natura della feature da implementare, in alcuni casi i test vengono scritti *prima* dell’effettiva implementazione di una specifica funzionalità (come previsto dal modello TDD), mentre in altri casi i test vengono ideati solamente *dopo* che gran parte del codice relativo alla feature è stato già scritto, al fine di validare il suo comportamento. In altri casi ancora può capitare di scrivere test per una libreria già esistente (e già testata in precedenza) al fine di estendere la *test coverage* di quella porzione di codice, sebbene ciò avvenga solitamente in preparazione ad un imminente *refactoring* dello stesso.

Il progetto utilizza la strategia del **controllo di versione distribuito**: tramite git ogni feature viene infatti sviluppata su un branch parallelo utilizzato internamente per lo sviluppo (*release branch*) e solo in seguito unita (*merge*) al branch principale, nel momento in cui essa viene validata e la sua introduzione approvata dal team (TODO: da sapere se è tutto il team o solo il lead dev che deve approvarla).

Ogni sviluppatore è libero di scegliere l’ambiente di sviluppo che preferisce, nonostante la maggiorparte dei membri del team utilizzi la suite di *IntelliJ IDEA*, grazie all’integrazione con le dipendenze necessarie al progetto che essa fornisce tramite i suoi plugin. (TODO: si può anche decidere di mettere questo nella sezione del team)

* Chiedere se hanno un documento con le cose da fare per questa release e lui che è il lead developer le assegna a ognuno di loro, o se ognuno sul gruppo “sceglie” cosa fare tra le cose da fare per la prossima release o un misto tra le due.
* Chiedere come funziona più precisamente l’inserimento di una feature all’interno del branch principale: è solo il lead developer che decide o lo si decide tutti insieme?
* Chiedere se esistono delle deadline (non avendo clienti, non dovrebbero averne di fisse, ma magari se le stabiliscono tra di loro per rimanere produttivi)