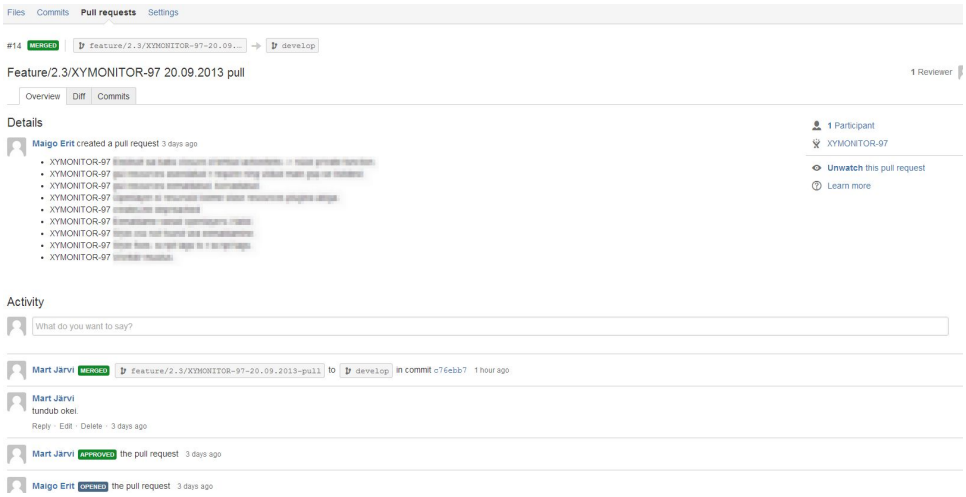
Lähtekoodi halduse ja ehitamise nõuded arendajale

Versioonihalduse kasutamine

- Tarkvara lähtekoodi halduseks tuleb kasutada https://source.smit.sise aadressil asuvad GIT repositooriumi (ligipääs antakse projekti põhiselt).
- Tarkvara versioonihalduses jälgitakse üldises mõistes **"git-flow"** protsessi (http://nvie.com/posts/a-successful-git-branching-model/) või **"feature-branche-workflow"** (https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow) protsessi.
- **"Git-flow"** lähenemist on sobilikum kasutada siis, kui tarkvara versioonide on aeglasem ja harvem ning versioonides toimuvad stabiliseerimisperioodid, samuti kui on vaja mitut erineva funktsionaalsusega versiooni pikaajaliselt toetada. Teine mudel sobib neile kes on rohkem automatiseerinud oma paigaldus ja tarneprotsessid ning kasutavad kas **"Continuous Delivery"** või **"Continuous Deployment"** töövooge.

- Iga JIRA pileti realiseerimise alguses loob arendaja JIRA abil (kasutades JIRA sees pileti juures käsku create branche) ***develop*** nimelisest harust endale vastava konvensiooni (vt. joonis) põhiselt ***feature/xxx*** nimelise haru kus arendust tehakse.
- Konfliktide vältimiseks peab kesksest harust järjepidevalt enda harusse muudatused mestima.
- Piletis realiseeritud lähtekoodi üleandmiseks SMIT-ile tuleb minna versioonihaldus keskkonna (https://source.smit.sise) vastava tarkvara ruumi ning valida sealt tab ***pull-requests*** ning luua uus ***pull-request***, kus tuleb määrata lähteharuks enda tehtud arendusharu ning lõppharuks keskne haru. Ülevaatajaks tuleb valida SMIT poolne arendaja ning sisuks täiendavad kommentaarid, mida silmas pidada antud tarnes (näiteks et muutus konfiguratsioon või baas vms).



- Tarne üleandmise eelduseks on, et üleantav kood vastab kõikidele kokkulepitud nõuetele, mis arenduse alguses on fikseeritud või on konkreetsed puudused toodud välja *pull-requesti* kirjelduses;
- Tarne loetakse vastuvõetuks, kui tarneharus oleva koodiga on toimunud vastavad tegevused:

    a. kood kompileerub SMIT ehitusserveris
    b. koodi staatiline analüsaator ei leia koodist vigu (kasutada Bitbucket Sonar pluginat code review raames või Bamboos Sonar taske)
    c. ühiktestid jooksevad läbi ilma vigadeta
    d. integratsioonitestidjooksevad läbi ilma vigadeta
    e. funktsionaalsed testid jooksevad läbi ilma vigadeta
    f. Koodi testide kattuvuse analüüs näitab et ei ole toodetud juurde ärikriitilist koodi, mis on testidega katmata
    g. arhitekt on tarne muudatustele teinud koodi analüüsi ja need heaks kiitnud
    h. kood on ilma konfiliktideta süsteemi poolt mestitud ***develop*** harusse (*pull-request* on täidetud)
    i. ***develop*** harust ehitatud versioon on paigaldunud arenduskeskkonda ja sinna on võimalik sisselogida
    j. Kõik tarnes sisalduvad ***commitid*** on seotud konkreetsete JIRA piletinumbritega kujul XXXX-YYY

- *Pull-request-i* kinnitamisel automaatselt kustutatakse vastav partneri poolt tehtud haru ära, kui mestimine on olnud edukas;
- *Pull-request*-ii võib SMIT tagasi lükata, kui seal esineb puudusi või alustada seal sees dialoogi puuduste kõrvaldamiseks (koodi ülevaatuse tegemisel lisatakse kommentaarid otse koodi ridade vahele);
- Arendaja peab arendusega seotud dokumentatsiooni kandma SMIT'i wikisse (https://wiki.smit.sise), vastava tarkvara ruumi (v.a paigaldusjuhend mis läheb koodi juurde versioonihaldusesse);
- Arendaja peab alati arendusi tegema JIRA (https://jira.smit.sise) piletite raames (iga koodimuudatus, mida soovitakse kesksesse versioonihaldusesse saata, peab sisaldama JIRA piletinumbrit) ning muutma nende staatuseid vastavalt arendusele.

Bamboo (CI/CD) kasutamine

- Igal tarkvaral on bamboos defineeritud 1 ehitusplaan, mis ehitab ennast "***develop***" või "***master***" plaani pealt automaatselt (haru valik sõltub, kumba protsessi kasutatakse koodi halduseks). Arendajad peaksid oma arendusi tegema feature harudes, mida automaatselt Bamboo on võimeline ehitama.

- Bamboo ehitusplaan ehitab koodi, teeb koodile staatilist analüüsi, võimalusel turvaanalüüsi, jooksutab testid ning Bamboo paigaldusplaan paigaldab lõpuks rakenduse määratud keskkonda.
- **"Git-flow"** puhul on reeglina ehitusplaan liidestatud "***develop"*** haruga ning paigaldatakse tulem arenduskeskkonda, testi ja toodangu jaoks versioonid tekivad harude pealt ("release" harud reeglina), mida paigaldatakse Bamboo kaudu käsitsi.
- **"Feature-branche-workflow"** protsessi puhul on ehitusplaan liidestatud "***master"*** haruga, mille tulemus paigaldatakse automaatselt sobivasse keskkonda. Võimalus on selle kõrvale luua ka täiendavaid harusid ja siduda neid konkreetse keskkonnaga.
- Toodangu keskkonda paigaldus tehakse reeglina käsitsi Bamboo sees ning sinna paigaldatakse sama tulem, mis läks testi.