5. Karakteristike i infrastruktura FOSS projekata

Za implementaciju FOSS projekata su potrebna određena tehnička znanja i veštine korisnika. Novi korisnici softvera imaju manje problema da savladaju nove veštine, kompjutere, softver, njegovu logiku i funkcionalnost, dok je više problema sa korisnicima koji imaju više iskustva i znanja, među kojima je dosta tehničkog osoblja. FOSS projekti koriste specifične i posebne alate koji podržavaju razvoj projekta i komunikaciju među programerima i korisnicima. Većina FOSS projekata nudi minimalan i standardan skup alata za upravljanje informacijama: web sajt, mailing liste ili forume, kontrolu verzija, praćenje grešaka, razgovor ("chat") u realnom vremenu. Takođe, karakteristika razvoja FOSS projekata je i to da programeri kao učesnici lako migriraju među različitim FOSS projektima.

Zajednički aspekti FOSS projekata su:

Zajednica – u FOSS projektima je bitan položaj unutar zajednice koji je se bazira na zaslugama doprinosa njegovom razvoju. Doprinos ne mora da bude samo količina napisanog programskog koda i može biti često povezana i sa preuzimanjem odgovornosti za delove projekta. Programeri vrlo često počinju učešće na projektu kao korisnici aplikacija, zatim identifikuju male promene koje mogu napraviti i ukoliko se promene prihvataju rade sve veće i veće promene i tako napreduju. Odgovornost je na pojedincu koji mora pregledati sve promene i osigurati da je promena dovoljno kvalitetna i da se inkorporira sa ostalim delovima koda u određenu funkcionalnost.

Liderstvo – u mnogim FOSS projektima postoji BD ("Benevolent Dictator"), osoba koja ima konačnu odluku i završnu reč. To je jedan ili nekoliko programera koji su se sa aspekta veštine izdvojili od običnih programera. Dobronamerni BD zapravo ne preteruje u "diktatorstvu" i oslanja se mnogo na iskusne i značajne programere koji imaju rezultate i iskustvo u određenim oblastima razvoja softvera. Neki FOSS projekti migriraju iz BD modela u demokratski model gde se odluke donose koncenzusom, tj. odlukom sa kojom se slaže većina, postoji diskusija za rešavanje problema.

Pronicljivost ("Forking") – je bitna karakteristika jer uvek postoji mogućnost da neko napravi kopiju softvera i započne sopstveni projekat na osnovu tog programskog koda. Primer ovakvog ra;vanja projekta jeste LibreOffice koji je formiran iz OpenOffice projekta. Ovakva situacija može nastati kao rezultat spora u razvojnom timu koji se ne može rešiti.

Komunikacija – FOSS projekti koriste zajedničke softverske alate za komunikaciju kao što je IRC (Internet Relay Chat) ili Redmine. Većina projekata ima različite mailing liste. Često se koriste i komunikatori za praćenje grešaka i sisteme za kontrolu verzija.

Planovi puta – FOSS projekti moraju imati definisan plan realizacije faza projekta i detaljan raspored izdanja softvera, grešaka, postupanja u slučaju grešaka, moguća poboljšanja. U manjim projektima je dovoljno sačiniti plan kao listu.

Izdanja – su vezana za trenutak u realizaciji projekta gde programeri žele da distribuiraju kod, tj. softver I njegovu dokumentaciju. Izdanja se rade na osnovu plana puta projekta. U jednom trenutku se razvoj "zamrzava" zbog testiranja koje se mora uraditi pre puštanja izdanja. Razvoj praktično ne prestaje, već se rade "grane", te je potrebno upravljanje različitim granama s obzirom da jedan broj programera radi na testiranju, a jedan deo na daljem razvoju kroz tzv. grane.

Skladišta/repozitorijumi – Veliki broj FOSS projekata koristi spremišta/skladišta/repozitorijume za čuvanje i organizaciju izvornog koda. Repozitorijum je mesto gde se skaldište softverski paketi i zvorni kod, odakle se isti može preuzeti, distribuirati i instalirati. Najpoznatiji repozitorijumi su: Launchpad, SourceForge i GitHub.

Pakovanje – FOSS projekti distribuiraju kroz izvorni kod koji treba da je u određenom formatu. Kod se mora preuzeti, po kompajlirati/interpetirati, ponekad i instalirati pomoću određenih

resursa, često i “ručno”. Veoma često se zbog toga formiraju binarni paketi da bi se olakšala instalacija i to ne moraju biti nužno kompajlirane datoteke. Paketi su unapred konfigurisani tako da mogu da budu instalirani bez direktnog formiranja iz izvornog koda.

Upstream/downstream – Razvoj FOSS projekata ne mora biti u potpunosti linearan. S obzirom da veliki broj ljudi radi i učestvuje na projektu istovremeno, formira se glavna grana bez obzira na sve iteracije (kaže se da projekat “teče uzvodno”). Kada se izvrši promena, ona se gura (“push”) uzvodno do glavne grane, tako da svi programeri uvek rade sa aktuelnom verzijom softvera. Ovi termini mogu biti vezani i za projekte koji su međusobno povezani, pa jedan projekat zavisi od realizacije nekog drugog, pa se izdavanje drugog projekta mora završiti pre izdavanja i puštanja prvog.

Kontrola verzija – U FOSS projektima učestvuje veći broj programera koji rade istovremeno pa je neophodno koristiti sisteme za kontrolu verzija, koji memoriše sve revizije izvršene na softveru radi kontrole promenau softveru. Ovi softveri obezbeđuju način komunikacije između članova tima, kao i način koordinacije promena u softvera. Postoji i mogućnost učenja na kontroli verzija.

Kontrola grešaka – u FOSS projektima je neophodno praćenje pronađenih grešaka (engl. “bug”-ova) i ispravki, tj. poboljšanja softvera koja su predložena. Alatka za praćenej grešaka je “tracker” i koristi se za praćenje zahteva za promenama unutar projekta. Zahtevi za promene mogu biti popravke grešaka, novi zahtevi za funkcijama, popravke tipa “zakrpa” drugih programera itd. Alat za praćenje grešaka omogućava da se problem prijavljuju, opisuju, dodeljuju prioriteti, reprodukuju, dijagnostifikuju, rešavaju i testiraju.

Životni ciklus grešaka u FOSS projektima: