Podpora rozhodovania vo voľne viazaných skupinách: štúdia stavu problémovej oblasti

SAŠO KISELKOV

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
skiselkov@gmail.com

Abstrakt. FOSS (Free and Open Source Software) je dnes jednou z najdynamickejšie sa vyvíjajúcich oblastí IT sveta. Poskytuje doposiaľ nevídanú flexibilitu v návrhu a integrácií riešení do rovnako profesionálneho ako aj neprofesionálneho sveta IT systémov. Táto nesporne veľmi atraktívna vlastnosť je naďalej podporená jeho častokrát extrémne nízkou cenou, čo umožňuje spoločnostiam nezáväzne so systémami experimentovať.

Napriek týmto veľmi veľkým výhodám, je samotný vývoj FOSS riešený akosi živelne. Vývojové tímy sa skladajú z dobrovoľníkov, ktorí vo svojom voľnom čase ten-ktorý projekt vylepšujú. Pritom sa jedná o častokrát veľmi voľne prepojený vývojový systém, pokrývajúci celý svet bez akéhosi koherentnejšieho riadenia. V takomto prostredí tradičné prostriedky na podporu riadenia, ktoré sa častokrát aplikujú s veľkým uspechom v uzavretých komerčných vývojových tímoch, jednoducho nefungujú. Potrebu podpory rozhodovania však nemožno len tak ignorovať - mnohé nádejné projekty končia nie kôli nedostatočnej motivácii alebo technickým možnostiam vývoja, avšak koli nekonzistentnosti krokov vedúcich k cieľu toho-ktorého projektu.

Cieľom tejto eseje je v prvom rade preskúmať procesy a princípy, na ktorých sa väčšia FOSS projektov a ich vývoj zakladá a pokúsi sa porovnať toto s tradičným princípom vývoja uzavretého softvéru. Ako ďaľšie sa pokúsi tieto princípy rozvninúť do metód a postupov, na ktorých by sa systémy pre podporu riadenia a rozohodovania pre FOSS projekty mohli zakladať, aby dokázali efektívne využiť obsiahnutý technický potenciál komunity.

Úvod

Ak povieme, že svet softvérového vývoja je neuveriteľne dynamicky sa vyvíjajúca oblasť, väčšina čitateľov, ktorí v tejto oblasti pracujú alebo sa ich nejakým spôsobom dotýka, bude súhlasiť. Neustále sa objavujú nové metódy na technickú prípravu

Manažment v softvérovom inžinierstve, október 2007, s. 1-9.

softvérových systémov. Skúšajú sa nové technológie a postupy vyjadrovania ich štruktúry, opisovania ich komplexnosti a taktiež samotnej organizácie ľudských zdrojov. Možno to ilustrovať pohľadom do histórie programovacích paradigiem . Kým v 50-tych a 60-tych rokoch sa značne využívala paradigma imperatívneho programovania priamo v nízkoúrovňových jazykoch, v rokoch 70-tych a väčšine 80-tych ju nahradila procedurálna paradigma vo vyžších programovacích jazykoch, ktorú zase v 90-tych rokoch takmer úplne nahradila paradigma objektovo-orientovaného vývoja. Táto rýchlosť sa môže zdať pomalá, avšak treba uvažovať ako veľmi sa od seba tieto prístupy líšia a akú vyjadrovaciu silu majú - v porovnaní so stavbárskym priemyslom je to akoby sa za približne 40 rokov prešlo od stavby nepálenou tehlou až ku oceľovo-kompozitným konštrukciám. A pokrok v oblasti softvérového vývoja napreduje aj naďalej stále rýchlejšie a rýchlejšie a otvára nám nové horizonty dosiahnuteľnosti.

Väčšina týchto snáh o technický pokrok je však sústredená okolo tradičného princípu vývoja softvéru v relatívne uzavretých skupinách. Jedná sa o tradičný vývojový model komerčného softvéru. Princíp tohto modelu je v podstate demand-driven, t.j. klient alebo skupina klientov si softvérový systém vyžiada a špecifikuje jeho presné vlastnosti. Softvérová firma potom projekt zrealizuje a dodá ako hotové riešenie, čím vývojový cyklus uzatvorí.

Vrámci týchto vývojových modelov existuje množstvo skúseností, či už z oblasti samotného riadenia ako aj podporovania riadenia. Existujú kompletné systémy a riešenia pre podporu týchto vývojových modelov, ktoré využívajú vlastností uzavretých vývojových tímov. Etablované sú napríklad plánovacie systémy, riadiace systémy, ticketovacie systémy a systémy pre podporu komunikácie v týchto tímoch. Zabezpečujú maximalizáciu produktivity tímu ako celku vďaka tomu, že sa môžu spoľahnúť na pomerne pevné a previazané ľudské prostredie.

Spomenuté tradičné komerčné vývojové modely pokrývajú širokú paletu zákazníckych požiadaviek, avšak nie všetky a nie všetky rovnako ideálne. V poslednej dobe sa v profesionálnej oblasti vývoja začína stalé viac presadzovať, ak ho možno tak nazvať, takmer až revolučný vývojový model. Tento je založený na priam utopistických myšlienkach:

- softvér by mal byť voľne šíriteľný
- každý používateľ by mal byť schopný ho upravovať a vylepšovať, aby si ho prizspôsobil na svoje požiadavky
- každý používateľ by mal mať možnosť svoj príspevok vrátit celej komunite

Jedná sa o priamy apel na vnútorné demokratické a humanistické hodnoty obsiahnuté v každom z nás, presadené do oblasti softvérového vývoja. Pred nie dlhou dobou by sme takýto model vysmiali so slovami, že nie je reálny. Avšak naša moderná informačná doba s rozmachom celosvetových komunikačných sietí mala omnoho väčší dopad na náš spôsob života a komunikácie, ako by ktorýkoľvek analytik bol pred nie dlhou dobou odhadoval. A bol to práve tento vývoj, ktorý podporil vytváranie celosvetových komunít, ktoré zdieľajú túto spoločnú ideológiu a chcú prispieť

k celkovému blahu aj takým nezjištným činom, akým je účastniť sa na kolaboratívnom vývoji softvéru bez nároku na odmenu.

Ako iste už mnohí tušia, projekty o ktorých sa tu píše sú známe pod spoločnou nálepkou FOSS (Free and Open-Source Software). Principiálnym vzorom takéhoto projektu je projekt GNU vedený Free Software Foundation [ref:FSF], ktorej zakladateľ založil spomínaný projekt v roku 1983. Tento projekt si kladie za cieľ vyvinúť úplne voľný operačný systém s otvoreným zdrojovým kódom. Odvtedy sa daný projekt vskutku rozrástol a je v podobe rôznych distribúcií GNU/Linux systémov naozaj schopný splniť svoj pôvodne veľmi ambiciózny cieľ. Medzičasom túto štruktúru vývoja už adaptovalo mnoho neziskových a ziskových organizácií, medzi ktorým možno snáď menovať:

- Internet Services Consortium (isc.org) - konzorcium, ktoré sa zaoberá vývojom plne funkčných softvérových Internetových štandardov, z pomedzi ktorých je snáď najznámejší BIND (Berkeley Internet Name Daemon - najčastejšie používaný softvér na realizáciu DNS systému) a sada ISC nástrojov pre DHCP (Dynamic Host Configuration Protocol - protokol pre automatickú konfiguráciu počítačov na sieti). Okrem toho sa ISC zaoberá vývojom Internetových štandardov a vyvíja neziskovú činnosť pre podporu existenie samoriadeného Internetu.
- The Apache Foundation - nadácia, ktorá sa zaoberá vývojom najčastejšie používaného web serverového softvéru známeho pod rovnakým menom Apache, avšak aj mnohých ďalších sieťových nástrojov.
- Sun Microsystems Inc. - komerčná korporácia, ktorá sa zaoberá predajom sieťových riešení, založené na systémoch a softvéri vyvíjané spomínanou firmou a komunitou dobrovoľníkov pod voľnými licenciami.

Týmto menovaním možno pokračovať dlho ďalej a určite sa v ních nájde mnoho spoločností, ktoré aj laici poznajú z počutia. Možno teda nesporne ilustrovať, že moderný a profesionálny svet sa rozhodol prijať FOSS riešenia a ich vývojový model.

Práve tieto vývojové modely sú však v doterajšom svete tradičných softvérových vývojových modelov veľmi nezvyčajné. Platí pre ne mnohé, ktoré pre klasické vývojové modely neplatilo. Napríklad FOSS projekty sú založené na dobrovoľnosti - nemožno nikoho nútiť, aby tú-ktorú úlohu vykonal. Vo väčšine taktiež neexistuje materiálnej odmeny, ale väčšina motivácie členov vývojového tímu spočíva v tom, že daný člen vie, že jeho príspevok do projektu je konštruktívny a pre komunitu prínosom.

Možno argumentovať, že mnohé tieto fakty hovoria proti vývojovému modelu FOSS, avšak realita sa ukazuje byť presne opačná - FOSS projekty sa práve z dôvodu dobrovoľnosti a absencie materiálneho rozmeru vyhýbajú hlavným demotivačným faktorom, známych z teórie manažmentu [1].

Špecifiká FOSS projektov

Okolo FOSS projektov a ich vývojového modelu sa však nachádza mnoho nevyriešených otázok, s ktorých jednou je ako čo najefektívnejšie podporiť ich interný riadiaci aparát. Z hore uvedených dôvodov, čím sú primárne odlišnosť od tradičného vývojové modelu uzavretej skupiny, na ne nemožno aplikovať klasickú metodológiu, nakoľko sa interný management FOSS projektov značne líši od komerčného managementového modelu. Rád by som spomenul aspoň zopár dramaticky odlišných čŕt, z ktorých by som nižšie rozvinul myšlienky ako asi možno tieto projekty efektívne podporiť v riadení a rozhodovaní. (Väčšina nižšie spomínaných čŕt vznikla viac-menej živelne - ľudia sa pri tvorbe týchto sociálnych systémov vážnejšie nezamýšlali nad ich efektívnosťou.)

Životný cyklus FOSS je jedna veľmi odlišná črta. Pri FOSS neexistuje akási striktná demand-driven politika nasadenia. Neexistujú preto fixné časové limity pre dosiahnutie míľnikov vo vývoji (a ak aj áno, tak ich nemožno brať veľmi definitívne) a taktiež absentuje väčšina krokov klasického životného cyklu: špecifikácia, návrh, implementácia, nasadenie, podpora. Životný cyklus typického FOSS projektu je na prvý pohľad jednoduchší:

- Úvodný hrubý návrh. Tu si zakladajúci člen, alebo členovia postavia akýsi cieľ a položia základné stavebné bloky pre jeho dosiahnutie. Robiť tento návrh príliš presným a stráviť nad ním veľa času je zvyčajne takmer isté pochovanie projektu. Nižšie je vysvetlené prečo tomu tak je.
- Implementácia hrubého návrhu. Nakoľko FOSS projekty sú založené na nemateriálnom uspokojení, je pre ne kriticky dôležité aby začali produktovať akýsi výsledok čím skôr vo svojom životnom cykle. Príliš dlhé trvanie návrhovej časti zvyčajne demotivuje všetkých jeho členov a efektívne ukončí projekt neúspechom. Toto je taktiež výrazne vyjadrené v klasickom príslovím tradovanom medzi FOSS vývojármi: "Release early, release often.". FOSS projekt musí svoje výsledky čo najskôr zverejniť komunite, kedže je to práve ona, ktorá nalieva do vývojového tímu hlavnú motiváciu pre pokračovanie projektu.
- Iteratívne vylepšovanie pôvodnej idei. V momente keď komunita príjme projekt a začne sa naozaj nasadzovať a používať je zvyčajne čas spracovať vstupy od komunity na vylepšenie projektu. Tento krok sa môže opakovať prakticky donekonečna a zaberá väčšinu času života FOSS projektu.

Absenciou časových limitov je prechod medzi týmito fázami nie je jasný a projekt sa môže nachádzať aj vo viacerých fázach zároveň. Hore uvedený životný cyklus nemožno brať ako definitívny - je založený na mojich pozorovaniach z projektov, v ktorých som sa zúčastnil aj na pozícií člena vývojového tímu aj vonkajšieho prispievateľa a teda člena "komunity".

Čo nás prináša ku ľudským zdrojom FOSS projektov a ich manažmentu. FOSS projekty sú veľmi voľne viazané systémy - väčšina väčších projektov pozostáva z viacerých úrovní ľudských zdrojov, ktoré sa líšia primárne svojou pracovnou angažovanosťou a šancou na pracovnú fluktuáciu (t.j. šanca odchodu z projektu, alebo pričlenenie sa k nemu):

- core project team, ktorý zvyčajne tvorí člen alebo členovia, ktorí projekt založili. Títo členovia väčšinou preberajú kombinované roly managerov, softvérových architektov a hlavných programátorov a vo všeobecnosti majú najmenšiu šancu pracovnej fluktuácie.
- core development team, teda hlavný tím vývojárov. Sú zodpovední hlavne za implementačné a architektonické črty projektu a podporujú core project team v rozhodovacích úlohach. Zvyčajne majú vyššiu mieru fluktuácie ako core project team.
- contributors, teda vonkajší prispievatelia do projektu. Toto sú najmenej "spoľahliví" a produktívni členovia FOSS projektu, čo sa však väčšinou vyvažuje ich veľkým počtom. Zvyčajne sa obmedzujú na vylepšenia implementačných častí projektu, programátorské úpravy, testovanie a ich príspevky musia väčšinou prejst cez schvaľovaciu "komisiu" tvorenú vyššími vrstvami riadenia v projekte. Netreba snáď podotknúť, že ich motivácia je zvyčajne veľmi nízka a fluktuácia veľmi vysoká.

Pozoruhodnou charakteristikou FOSS projektov je, že hore uvedené úrovne "zaviazanosti" projektu môžu členovia často meniť počas existencie projektu. Mimo spomínaných vrstiev je posledná vrstva - používatelia projektu, ktorí o sebe nedajú nič vedieť. Tí sú však, z pohľadu projektu jedine dôležitý pre štatistické údaje a neučastnia sa jeho priame vývoja.

Ako podporiť FOSS projekt v rozhodovaní a riadení?

Vzhľadom na špecifické vlastnosti FOSS projektov je preto treba navrhnúť nové pravidlá a nové metodológie pre ich efektívnu podporu pri riadení a rozhodovaní. V tradičných systémoch túto rolu preberali systémy niekedy označované ako GDSS (Group Decision Support System, [3]), čo je kompozícia hardvérových a softvérových prvkov určených na efektívne podporovanie rozhodovacieho a riadiaceho procesu v manažmente softvérového projektu. Môžu ho tvoriť napríklad:

- telekonferečné systémy pre podporu komunikácie medzi členmi tímu, súčasťou ktorých sú napríklad moderačné podporné prvky, ktoré pomáhajú "moderátorovi" konferencie riadiť výmenu myšlienok, aby konferencia efektívne využívala jej poskytované prostriedky.
- datové úložiská a repozitáre, pre uschovávanie nápadov a návrhov, ktoré tímy počas konferencií naakumulujú, ako aj systémy pre ich opätovné vyvolanie.
- systémy pre kontrolu dodržiavania termínov a postupu v projekte.

Všetky tieto systémy sú zvyčajne dodávané v ako jeden kompletný balík služieb, ktorý potom organizácia využíva na svoje interné riadenie. Hore spomenuté prostriedky ako také možno považovať univerzálne - v princípe sa snažia len o to sprostredkovať čo najefektívnejšiu komunikáciu medzi členmi tímu. Typickým problémom je, že niektorí členovia sú autoritatívni a teda preberajú akúsi rolu "hovorcov", čím podvedome utlačia slabších (hoci potenciálne inovatívnych) členov do úzadia [3]. GDSS sa snažia tieto efekty potlačiť zavádzaním striktnejších metód kontroly komunikácie medzi členmi a taktiež sa snažia svojímy datovými repozitármi o to aby sa "myšlienky nestrácali", keďze to je jeden z hlavných faktorov nevyužívania ľudského potenciálu. Voľba správneho nástroja a jeho metódy použitia je pritom kriticky dôležitá, nakoľko musí byť vyhovujúci spôsobu myslenia a riešenia problémov v danom tíme, a taktiež určuje akým spôsobom bude tým komunikovať, z čoho vyplýva teda aj aké výsledky bude dosahovať [2].

Spomenuté komponenty GDSS možno principiálne deklarovať za priamo aplikovateľné aj na FOSS projekty - FOSS projekty trpia podobnými problémami v oblasti riadenia, teda napríklad, že sa myšlienky počas skupinových diskusií stratia, alebo že silnejší členovia "prekričia" slabších. Problémom však ostáva, že kompletné GDSS riešenia sú vo väčšine prípadov pre FOSS projekty príliš ťažkopádne. Pre ich efektívne použitie potrebuje FOSS projekt, aby spomenuté riešenia boli, pokiaľ možno, čo najviac "odľahčené" od prebytočného balastu. FOSS projekty sa taktiež väčšinou rozprestierajú cez veľké geografické oblasti, preto treba riešiť ich kolaboráciu na úrovni Internetu a datových sietí, ktoré geografickými obmedzeniami netrpia. Ideálny podporný systém pre FOSS projekt preto asi bude pozostávať z:

- Webovej aplikácie, ktorá funguje z ľubovoľného miesta na svete a je použiteľná aj na komoditnom hardvéri. Táto môže poskytovať členom projektu portál ku hlbším službám systému.
- Distribuovanej databázy, ktorá nevyžaduje veľa prostriedkov pre správu. Typickým predstaviteľom tohto sú mailing-listy, ktoré sú vo FOSS projektoch už dnes veľmi intenzívne využívanym prostriedkom. Ich nevýhodou je však veľmi nízka rýchlosť komunikácie.
- Ticketový systém. Tento systém pomáha sledovať problémy v projekte vedúcim projektu, avšak narozdiel od typického komerčného ticketového systému, ktorý potom pridelí vyriešenie problému nejakému jeho členovi, by mal takýto ticketový systém podporovať dobrovoľnú spoluprácu na vyriešení problému. Mal by v sebe integrovať funkčnosť komunikačného, testovacieho a sledovacieho prostriedku - jednotliví členovia tímu, ako aj členovia komunity, vidia ktoré časti kto práve rieši a svoje medzivýsledky spätne posiela do ticketového systému, aby mohol s riešením problému prípadne pomôcť ďalší člen tímu alebo komunity.

Aktuálny stav FOSS GDSS systémov

V tejto časti by som rád pozrel na aktuálny stav GDSS systémov, ktoré sú vhodné pre FOSS projekty.

FOSS komunita si podvedome uvedomila potrebu pre GDSS systémy už pred dlhšou dobou a začala s vývojom špecifických riešení, ktoré si brali na mušku špecifické problémy pri podpore vývoja FOSS projektov. Hlavne by som rád spomenul dve spomedzi najväčších: SourceForge a Savane.

SourceForge

SourceForge systém je používaný na stránkach projektu SourceForge.net a jedná sa uzavretý systém. Jeho využitie je síce voľne dostupné komukoľvek, avšak zmeny v ňom nie sú možné. Aj tak SourceForge projekt obsahuje mnohé funkcie mierené práve na podporu vývoja FOSS projektov. Medzi jeho hlavné časti patria:

- kontrola a riadenie členov vývojového tímu.
- verziovaný repozitár zdrojových kódov projektu s podporou pre CVS ako aj Subversion.
- integrované vytváranie mailing listov pre tímovú komunikáciu.
- bugtracker/ticketový systém pre komunikáciu s komunitou ohľadne problémov a požadovaných zmien, plne integrovaný so sledovaním vývoja daného problému.
- priestor na webové stránky a verejnú prezentáciu projektu .
- riadenie osobných profilov členov - každý člen sa môže na serveri prezentovať a deklarovať na akých projektoch spolupracoval a projekty môžu poriadať "náborové" akcie, čím sa môžu projekt aj potenciálni vývojári lepšie priblížiť.

Veľmi zaujimavým rysom projektu SourceForge.net je jeho prístup ku existencií samotných projektov. Registrácia používateľského účtu na serveri, ako aj projektu je úplne bezplatná a pozoruhodne, staré projekty nemožno zmazať. SourceForge.net sa snaží aj opustené projekty zachovať, aby maximaloval šancu, že zaujimavý ale zanechaný projekt niekto postrehne a prevezme a takto podporí ku úspešnému pokračovaniu, čím benefituje komunita.

Medzi hlavné problémy SourceForge.net projektu patrí práve to, že sa jedná o projekt uzavretý a teda nepodlieha samotným FOSS pravidlám, ktoré sa snaží podporovať. Ak teda nejaký používateľ SourceForge.net projektu nie je spokojný s tou-ktorou jeho črtou, nemôže s tým žiaľ nič spraviť.

Savane

Projekt Savane je systém etablovaný samotnou FSF pre GNU projekt. FSF kedysi používala SourceForge systém, ktorý sa však jeho materská organizácia rozhodla

uzavrieť. FSF sa preto naďalej rozhodla, že bude pokračovať samostatným vývojom v poslednej verejnej verzií SourceForge systému, čím vytvorila projekt Savane.

Podobne ako SourceForge, sa jedná sa o rozsiahlý webový systém, primárne používaný na stránkach savannah.gnu.org a gna.org, ktorý slúži na účel poskytnutia centrálneho úložiska ľubovoľného softvérového projektu a jeho podpory zo strany vývojárov a komunity postredníctvom poskytovania rôznych komunikačných prostriedkov. Samotný Savane systém je vlastne len komunikačný "engine" spojený s databázou zdrojových kódov a ostatných súčastí projektu. Je dostupný bezplatne komukoľvek za podmienok liberálnej licencie.

Hore spomínané systémy majú nedostatky v oblasti rýchlej komunikácie. Ani jeden z nich nepodporuje akúsi formu konferenčných diskusií, čo si vývojový tím väčšinou supluje pomocou iných systémov, ako napríklad IRC alebo Jabber. Taktiež je ich problémom pre oblasť podpory riadenia a rozhodovania, že neboli pôvodne pre ňu zamýšľané. Riadiace funkcie boli do nich pridané až dodatočne. Prvotné systémy boli v podstate integrované repozitáre zdrojových kódov.

Záver

Z predošlých odstavcov by som rád zhrnul svoje myšlienky následujúcim spôsobom.

Ako možno efektívne podporiť FOSS projekt pri rozhodovaní a riadení? Primárne je potrebné si uvedomiť rozdielu medzi FOSS projektmi a komerčnými projektmi. Jedná sa o dva rozdielné prístupy, k tomu istému problému - ako čo najlepšie spojiť ludí, aby vytvorili čo najlepší technický výsledok. Na toto treba príslušné technické prostriedky, ktoré musia oplývať týmito vlastnosťami:

- Musia byť ľahké (lightweight) a lacné. Drahý a komplikovaný systém si nezisková FOSS komunita nemôže dovoliť.
- Musia podporovať ideu FOSS a spoluprácu s komunitou dobrovoľníkov. Uzavreté vývojové tímy nedokážu dodať kvalitný FOSS produkt v reálnom čase.
- Musia byť sieťovo distribuované, aby kopírovali voľné viazanie tímov tvoriacich FOSS projekty.

Dnešné nástroje, ktoré vznikli viac-menej živelne v oblasti podpory vývoja softvérových systémov, poskytujú aj čiastočné funkcie pre podporu riadenia, avšak nevyužívajú ešte plný potenciál moderných webových technologií pre túto funkciu.

Použitá literatúra

1. Barleson J.A.: An organizational solution to employee fatigue. *Proceedings of the 8th annual ACM SIGUCCS conference on User services*, (1980) 18-20

2. Fox T.L, Spence J.W.: The effect of decision style on the use of a project management tool: an empirical laboratory study. *ACM SIGMIS Database*, Vol. 36, Issue 2, (Spring 2005), 28-42
3. Kraemer K.L., King J.L.: Computer-based systems for cooperative work and group decision making. *ACM Computing Surveys (CSUR)*, Vol. 20, Issue 2 (June 1988), 115-146

Annotation

Decision making support in loosely coupled groups: a problem domain study

Supporting decision making in loosely connected groups: a study of the status in the given area
FOSS (Free and Open Source Software) is today among the most dynamically developing areas in the world of IT. It offers unprecendented flexibility in design and integration for both professional and non-professional IT systems. This undoubtedly very attractive feature is further enhanced by the often very low price, which allows companies to quickly experiment and prototype new systems.
Despite these great advantages, the development of FOSS is done kind-of „in the wild“. Development teams are comprised of volunteers which enhance a given project in their spare time. This forms a loosely connected system without a significantly coherent management structure. In such an environment traditional decision-making support mechanisms, which are applied with great success in closed commercial development teams, simply don't work. The need for decision-making support cannot, however, be simply ignored – many great projects end not because of insufficient motivation or because of technical issues, but instead because of inconsistent steps taken to achieve the given goal.
This essay tries to research the processes and principles which govern FOSS projects and their development, and tries to compare this to the traditional principles usually found in traditional closed-team developed software. Next, it will try to extend these principles into methods and processes based on which systems for supporting decision-making in FOSS projects could be based, so that they can efficiently exploit the technical potential contained in the community.