Plánovanie – žiadna dogma, iba prostriedok

MARTIN ĽUDVÍK

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
ludvik05[zavináč]student[.]fiit[.]stuba[.]sk

Abstrakt. Na softvérových projektoch sa podieľajú ľudia so spoločným cieľom - uspokojiť potreby zákazníka. Aby sa však zo skupiny ľudí stal tím, potrebujú sa spolu koordinovať a kooperovať. Ako čo najefektívnejšie usmerniť ich správanie? Ako predpokladať ich produktivitu, odhadovať čas riešenia jednotlivých problémov, distribuovať ľudské zdroje medzi paralelné procesy vývoja? Na riešenie podobných problémov existuje rada metodík a nástrojov. V tomto článku sa však nebudeme zaoberať tým, ako ich použiť. Radšej sa pokúsime odhaliť motiváciu, ktorá za nimi stojí. Pozrieme sa na prínos, ktorý v sebe ukývajú, a spojitosti, ktoré môžu byť poučné. Pokúsime sa presvedčiť čitateľa o tom, že plánovanie nie je doktrína, ale iba akýsi aparát intelektu, ktorý má poslúžiť pre formulovanie vlastných názorov a zásad.

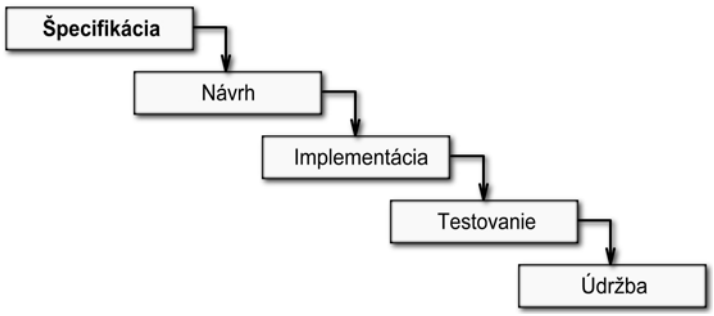
Úvod

Kam sa len pozrieme, môžeme vidieť majestátne výtvory ľudstva - či už ide o skvosty architektúry, gigantické plavidlá alebo drobné mikročipy. Väčšina z týchto diel by nevznikla bez prvotného plánu. Rovnako v softvérovom inžinierstve sa stretávame s potrebou explicitne vyjadrovať cestu k svojim cieľom. Do istej miery sa nám to aj darí a zásluhu na tom majú ľudia zvučných mien, ako je Booch, Sommerville a ďalší. Akokoľvek, softvérové inžinierstvo je veda pomerne mladá a ako taká pôsobí dojmom nestálosti a nepretržitého vývoja. Inak povedané, neexistuje najsprávnejšia metodika pre plánovanie. Tá sa rôzni od firmy k firme ba i od projektu k projektu. Čo však existuje, sú odskúšané postupy a odporúčania, ktoré by nám mali poslúžiť ako predloha pri modelovaní našich vlastných procesov spojených s vývojom softvéru.

Čomu nás učia modely

Navzdory faktu, že exaktné definovanie postupov pri plánovaní (a vývoji vôbec) softvérového systému je relatívne vágne, a teda rôzne interpretovateľné, určite sa môžeme zhodnúť v niektorých bodoch. Po prvé, zložitý problém je dobré rozdeliť si na menšie podproblémy, tie potom riešiť samostatne a nakoniec previesť ich syntézu.

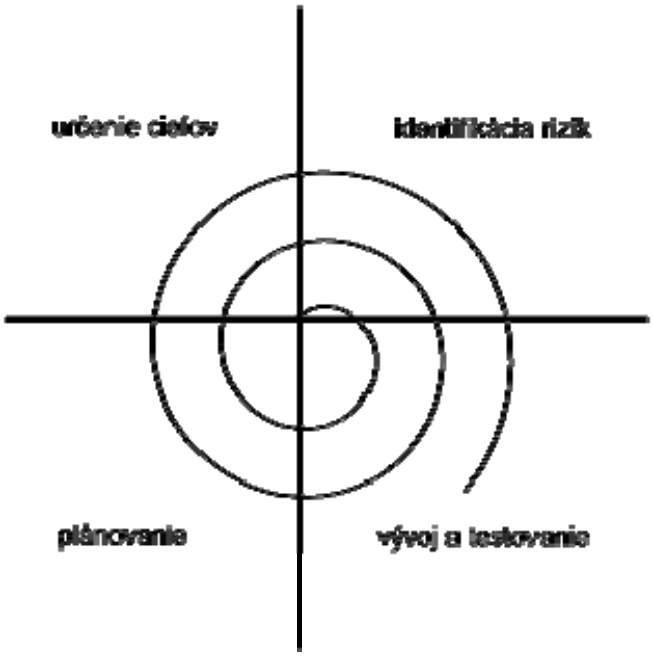
V tomto duchu sa dostávame k idei životného cyklu vývoja systému. Historicky prvým opísaným je nám už notoricky známy vodopádový model (pozri Obr. 1). Od svojho vzniku podlieha kritike v náväznosti na jeho statickosť či nízku flexibilitu a v reálnych projektoch je prakticky nepoužiteľný. Neznamená to však, že nám nemá čo dať. Pozrime sa na neho ako na akúsi abstrakciu, odrazový mostík pre všetky ďalšie modely. Pomáha nám uvedomiť si členenie procesu vývoja. Definuje a pomenúva jednotlivé etapy, vymedzuje ich význam a dáva ich do časových súvislostí. Nie je dôležité, či v jeho zobrazení vedú šípky iba smerom vpred, alebo aj ku predchádzajúcim etapám. Nie je dôležité ani to, či je rozdelený na 5 alebo 6 etáp - beztak ho nikto nepoužije absolútne v súlade s jeho pravidlami. Pre mňa predstavuje jeho prínos myšlienka dekomponovať vývoj na viacero častí, ktoré bude možné zvlášť monitorovať, kontrolovať ich kvalitu, plánovať, ďalej dekomponovať atď. Vodopádový model je významným míľnikom, pri ktorom sa môžeme zamyslieť nad otázkami ako je vhodná granularita plánovania alebo určenie kontrolných bodov vo vývoji.



Obr. 1. Vodopádový model

Hľadanie nových techník vývoja softvéru mnohokrát pripomína hľadanie „zlatej strednej cesty“. Na jednej strane je nekontrolované, nijako neusmerňované programovanie a na druhej striktný, ortodoxný vývojový model (ako napr. už spomínaný vodopádový model alebo V-model). Domnievam sa, že ideálne riešenie je kdesi uprostred a my ku nemu konvergujeme striedavo z jednej a druhej strany. Nasvedčuje tomu aj postupné prepracovanie sa ku iteratívnemu modelu vývoja, ktorý predstavuje „recept“ pre správne miešanie programovania a manažmentu softvéru. Už len pohľad na jeho symbolické zakreslenie formou špirálového modelu (pozri Obr. 2) nám napovedá, že bude v niečom významnom odlišný od vodopádového modelu. Kým vodopádový model je možné zakresliť akoby lineárne, iteratívny sa rozprestiera na ploche. Mohli by sme povedať, že má o jeden rozmer viac. A skutočne! Ak sa zamyslíme, tak vodopádový model delí vývoj na etapy z pohľadu manažmentu a úplne ignoruje kompozíciu samotnej funkčnosti vytváraného systému. Iteratívny vývoj sa snaží zachovať tento systém delenia, ale navyše zavádza aj rozmer druhý a tým je práve spomínané nabaľovanie funkcionality. Iteratívny model nás učí, že pozerať sa

máme vždy skrz viacero kritérií a keď už vyčerpáme všetky možnosti, musíme otvoriť „ďalšiu dimenziu“.



Obr. 2. Špirálový model

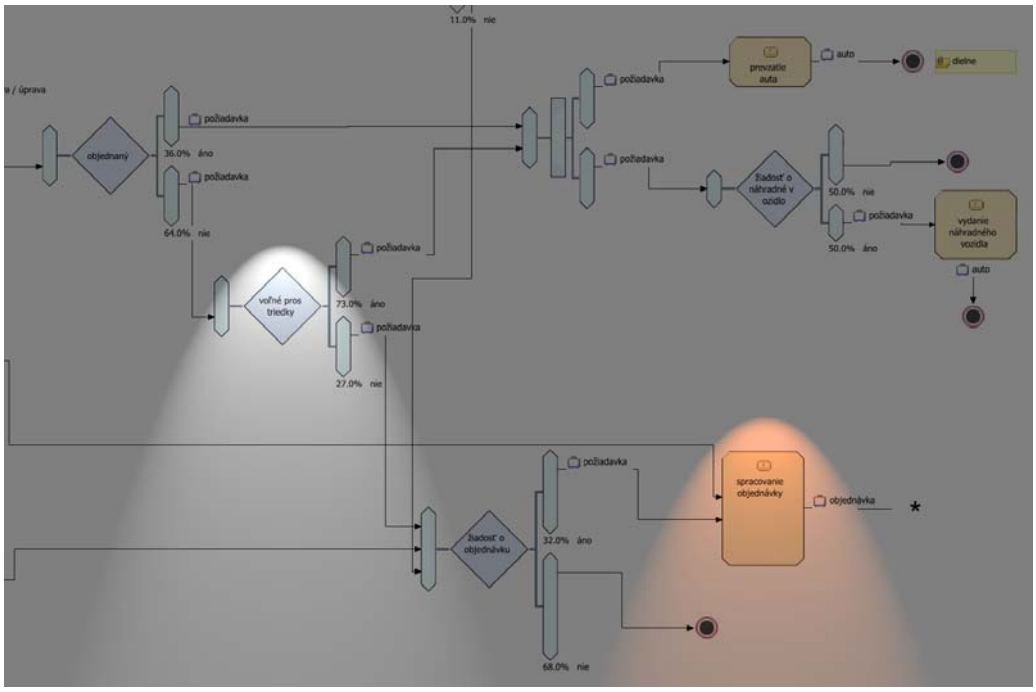
Ako som už povedal, naše vývojové paradigmy pomaly konvergujú ku kompromisom medzi dvoma extrémnymi tábormi. Vyzerá to tak, že iteratívny model je dostatočne flexibilný, aby mohol byť uspokojivo nastavený pre väčšinu projektov. Z tohto pohľadu sa teda už nemá moc čo vylepšovať. Predpokladám, že iteratívny model je vyvrcholením našich snáh v oblasti životných cyklov. Ako teda ďalej, keď sme už na vrchole? Kde hľadať nové možnosti rozletu? Odpoveďou je agilné programovanie. Ono je tou ďalšou dimenziou, o ktorej sme hovorili. Zavádza do vývoja úplne nové prvky, ako je integrácia budúceho používateľa do procesu tvorby softvéru alebo programovanie v pároch. Avšak vyjadrovať sa ku nemu bližšie by mohlo byť predčasné. Ide totiž o trend, ktorého súčasťou sme práve my, a preto nemôžeme byť dostatočne objektívni. Je pozoruhodné, ako jednoducho sa ľudia zhodnú na veciach, ktoré sú prežitkom (opäť uveďme ako príklad vodopádový model). Zastávať ale názor o procesoch, ktorých sme sami súčasťou, nemusí byť vždy najšťastnejšie. Nechajme preto agilné programovanie vyzrieť, a keď už nebudeme vedieť ako ďalej, príde niekto ďalší, kto nám predostrie novú dimenziu.

Čo ukrývajú škatuľky

Plánovať iba na základe modelov samozrejme nestačí. Každá etapa môže byť (a celkom určite bude) ďalej rozbitá na menšie časti. Prvá otázka, ktorú si položíme, znie: „Kedy začať plánovať jemnejšie detaily vývoja?“. Ideálne by bolo urobiť to hneď na začiatku projektu. To však nie je možné pre vysoký stupeň neurčitosti, ktorý sprevádza túto fázu. Navyše plán by bol príliš statický a neodrážal by realitu. Pokiaľ by bol navrhnutý optimisticky, čoskoro by sa dostal projekt do časového sklzu. Ak by bol naopak pesimistický, dochádzalo by ku plytvaniu časom na miestach, kde to vôbec nie je potrebné. Plány teda zjemňujeme neprestajne počas celého vývoja a vždy pre tie etapy, ktoré budú v blízkej budúcnosti nasledovať (viď. [1]). Plánovať jednu etapu vopred je výhodné preto, aby sa načas alokovali a pripravili zdroje na ňu potrebné. Inak sa môže stať, že ľudské zdroje už budú síce voľné, ale bude chýbať napr. technické vybavenie [2].

Druhá otázka môže znieť: „Nakoľko jemné majú byť plány?“. Správne by sme sa však mali pýtať: „Čo svojimi plánmi vlastne sledujeme?“. Sledujeme čas? Sledujeme potrebu zdrojov? Povedzme, že by sme rozložili procesy na také (čo najväčšie) podprocesy, medzi ktorými teoreticky nedôjde k presunom ľudských zdrojov. Domnievam sa, že takýmto spôsobom navrhneme plán, ktorý bude účinný pri predpokladaní potrebných zdrojov. Či bude aj dostatočne presný pre odhad potrebného času, to ponechávam na zváženie čitateľovi. Podstatné je uvedomiť si, aké dôležité je robiť veci pre ich pravú príčinu. Ak plánujeme iba preto, lebo sa to má, tak sa naša snaha pravdepodobne minie účinku. Radšej sa zamerajme najprv na to, čo chceme vyzistiť, a až potom podľa požiadaviek aplikujme známe postupy.

Po tom, ako si všetko starostlivo premyslíme, prichádza na rad vhodná reprezentácia plánov. Čas ukázal, že výborným riešením je vizualizovať ich pomocou grafov. Výhodou grafu je fakt, že rozvíja informáciu do priestoru. Už Kant vo svojej Kritike čistého rozumu naznačil, že ľudské vedomie je ohraničené svojimi vnemami, a teda priestorom a časom. Preto sú mu tieto dva formy vnímania také blízke, oveľa bližšie, ako zlievajúci sa text. Na to netreba zabúdať pri interpretácii našich myšlienok iným. Nazdávam sa, že experimentovanie s pridaním tretieho rozmeru alebo časovej zložky do statických grafov by mohlo byť prinajmenšom zaujímavým námetom pre experimentovanie. Istým spôsobom sa s týmto prístupom môžeme stretnúť vo forme simulácií diagramov toku dát či iných diagramov. Problémom, ktorý by tu hádam nastal, je príliš veľa animácie na veľkej ploche súčasne. Manažér v roli diváka nemôže byť schopný sledovať toľko informácií naraz. V tomto by mali informatici požiadať o pomoc kolegov z iných disciplín, ktoré sa na prvom mieste zaoberajú divákom, a využiť prostriedky ako je štronzo, či metaforu svetelnej techniky. Na Obr. 3 vidíme jednu z mojich predstáv takejto simulácie. Počítač vyhodnocuje miesta, ktoré by mohli manažéra potenciálne zaujímať. Tie zvýrazní pomocou svetelných efektov a urgenciu daných javov môže ešte zvýrazniť farbou nasvietenia.



Obr. 3. Metaforma svetelnej techniku v simulácii

Podporné nástroje

Snaha zjednodušiť a zefektívniť proces plánovania v čo najväčšej miere viedla k vytvoreniu robustných nástrojov. Väčšinou ide o softvérové systémy, ktoré podporujú tvorbu rôznych diagramov, Ganttových grafov, sledovanie termínov atď. Tým, kto ale v konečnom dôsledku rozhoduje, je predsa len manažér. Ak uvážime mieru neurčitosti a množstvo rôznorodých rizík, ktoré musí brať do úvahy, môžeme konštatovať jediné - manažér by musel byť stúpencom pansofie, aby bol schopný produkovať optimálne rozhodnutia. Preto existujú pokusy preniesť časť rozhodovania do rúk počítačov. Dôkazom toho je článok [1], ktorý hľadá súvislosti medzi automatizovanými robotmi vyvíjanými NASA a manažmentom softvéru. Ich spoločnými črtami sú obmedzené zdroje a čas, ako aj množina úloh s rôznou prioritou. Na plánovanie sa potom autor článku pozerá z pohľadu simulácie, v ktorej má každý člen vývojového tímu nadefinované svoje parametre (učenlivosť, rýchlosť...). Na základe týchto parametrov a pravdepodobnostného zohľadnenia rôznych rizikových faktorov prebieha výpočet, ktorý v pravidelnom rytme (povedzme každý týždeň) prepočítava najvhodnejšie strategické rozhodnutia. Článok však už nepojednáva o tom, akým spôsobom ľudí ohodnocovať. Určite nie je ľahké nadefinovať každému zamestnancovi mnoho parametrov a následne vytvoriť systém, ktorý z týchto parametrov vyvodí príslušné

závery. Tu by nám mohla pomôcť umelá inteligencia. Mojim nápadom je zautomatizovať sledovanie členov tímu a porovnávať dopad odpozorovaných parametrov na výsledok konkrétnych úloh. Pomocou evolučných algoritmov alebo neurónových sietí by tak bolo možné zistiť drahocenné súvislosti a tie ďalej použiť pre účely predikcie vývoja softvéru. Navyše, je tu aj druhá potenciálna možnosť, ako využiť zozbierané údaje. Odpozorované parametre by mohli byť premietnuté vo forme vektorov do n-rozmerného priestoru. Správne volený rez takéhoto priestoru nám môže veľa prezradiť o psychologických podobnostiach jedincov, ich pracovitosti, časovej flexibilite atď. To sa dá následne využiť napr. pri zgrupovaní členov tímu na jednotlivé podúlohy a pod.

Záver

Softvérové inžinierstvo je úžasná veda, ktorá má však jeden háčik – chýba jej určitosť. Nie je možné (a ani žiadúce) slepo využívať jej nástroje bez hlbšieho zamyslenia sa nad ich pozadím. Neberme tieto nástroje ako konečné a nemenné. Naopak, naučme sa extrahovať z každého to dobré a experimentujme s ich kombinovaním. Pritom nesmieme zabudnúť, že nič nemá byť samoúčelné a niekedy prostá skica prevyšuje aj ten najsofistikovanejší model.

Použitá literatúra

1. JOSLIN, D. - POOLE, W.: *Agent-based simulation for software project planning*. In: Proc. of the 2005 Winter Simulation Conference, M. E. Kuhl a kol. (Eds.), Orlando, Florida (2005), 1059 - 1066
2. LEHTOLA, L. a i.: *Strengthening the link between business decisions and Long-term product planning in software product companies*. Int. Requirements Engineering Conference, (2007), 153 – 162
3. RETTIG, M. - SIMONS, G.: *A project planning and development process for small teams*. Communications of the ACM, Vol. 36 (1993), 45 - 55

Annotation

Planning – not a dogma, just a tool

People work on software projects with one objectivity – to order customer`s needs. But before we can consider group of people to be a team, they need to coordinate and cooperate each other. How should we control their behavior most efficient? How can we predict their productivity, estimate time for solving particular tasks, distribute people resources between parallel processes of development? There are many tools to help solve this problems. But we will not discuss, how to use them. Instead, we will try to find the motivations, that stands behind this techniques. We will take a look on the contribution, that is hiding inside them, and relations, that may be interesting. We will try to convince the reader, that planning is not a doctrine, but some kind of intellect`s apparatus, which should serve to formulate own opinions and principles.