Agilní metody řízení projektů

Zuzana Šochová
Eduard Kunce

Agilní metody řízení projektů

Vyšlo také v tištěné verzi

Objednat můžete na
www.computerpress.cz
www.albatrosmedia.cz

Zuzana Šochová, Eduard Kunce
Agilní metody řízení projektů – e-kniha

Zuzana Šochová, Eduard Kunce

Agilní metody řízení projektů

Computer Press
Brno
2019

Obsah

Úvod 10

Zpětná vazba od čtenářů 12

Errata 12

Část I – Filozofie agilních a lean metod 13

Agilní a lean metody 15

Co si představíte pod slovem „agilní“? 15

Manifest agilního vývoje softwaru 15

Co je to „lean“? 25

Co je to Scrum? 26

Co je to Kanban? 26

Co je to Scaling (škálování)? 26

Agilní transformace 29

Vývoj softwaru 29

Jaké jsou nejčastější důvody pro přechod na agilní metody? 31

Business Agility 35

Agilní organizace a agilní leadership 35

Část II – Popis metod, procesů, praktik a artefaktů 37

Agilní slovníček 38

Lidi a vztahy 43

Scrum Master 43
Product Owner 47
Self-organized tým 49
Multifunkční tým 51
Scrum tým a Development tým 53
Zákazník 55
Role manažera 57
Role projektového manažera 58

Praktiky a artefakty 61

Sprint Goal 61
Prioritizace 63
Product Backlog Item 67
Sprint 69
Product Backlog 73
Sprint Backlog 75
User Story 77
INVEST kritéria 79
Epic a Témata 83
Definition of Done 84
Dobrá vizualizace 87

Scrum Tabule 87
Kanban Tabule 89
Kdy potřebujeme nástroj? 89
Ohodnocení 93
Relativní jednotky versus odhad času 95
Planning Poker 99
Rychlost 103
Meetingy a aktivity 107
Standup / Daily Scrum meeting 107
Retrospektiva 111
Backlog Refinement 119
Backlog Grooming 119
Sprint Planning 121
Sprint Review 125
Agilní programování – artefakty 127
Pair Programming 127
Mob Programming 128
Review 128
Sdílený kód 128
Coding Standard 129
Jednoduchý design 130
Continuous Integration 131
Test Driven Development – TDD 132

Refactoring 133
Časté releasy 134
Žádné přesčasy 134
Společný cíl 134
Technický dluh 135
Scrum 1-1 139
Příprava Scrumu 141
Scrum pro více týmů – Scaling 143
Scrum of Scrums a další synchronizační meetingy 144
Extreme Programming 1-1 147
Kanban 1-1 151
Jak funguje tým 155
Tuckman's Group Development Model 155
Pět důvodů, proč týmy nefungují 158
Smlouvy a formy spolupráce 163
Fixed Time, Fixed Price 163
Agilní smlouva 164
Otázky na závěr 167

Část III – Implementace, transformace, změna 171

Průvodce agilní změnou 173

Proč agilně 173

Bude to zig-zag 175

Co máme dělat, když... 177

Co máme dělat, když jsme malá firma? 177

Co máme dělat, když jsme středně velká firma? 178

Co máme dělat, když jsme velká firma? 178

Co máme dělat, když děláme jen migraci? 181

Co máme dělat, když jsme projektově organizovaná firma? 183

Co máme dělat, když máme distribuovaný tým? 184

Jak stavět Product Backlog 187

Product Discovery 187

Story Mapping a Project Chartering 189

„Best Practices" Scrumu 192

Nejčastější nepochopení Scrumu 194

Agilní organizace 197

Ohodnocování a metriky 201

Část IV – Příklady 203

Průvodce agilní transformací – co budete při zavádění Scrumu potřebovat 205

Ukázka Backlogu 207

Část V – Případové studie 209

Jak změnit styl práce 211

Menší firma – Cesta k agilní organizaci 211

Kontext 211

První fáze: Ostrůvky (2005–2006) 211

Druhá fáze: Agilní týmy: (2006–2008) 212

Třetí fáze: Agilní organizace: (2008–2010) 213

„Spotify Model" 215

Mezinárodní korporace 216

Ambiciozní startup 217

Závěr 219

O autorech 220

Zuzana Šochová 220

Eduard Kunce 221

Ostatní kontakty: 222

Agilní asociace 222

Agile Prague Conference 222

Reference: 224

Úvod

Dostává se vám do ruky publikace o agilních metodách. Snažili jsme se čtivou formou popsat všechna možná zákoutí a nástrahy, které vás při přechodu na agilní metody mohou potkat. Kniha nepopisuje jen teorii, co to agilní metody jsou, jak funguje Scrum Proces a Kanban a jak by jednotlivé artefakty těchto procesů měly správně vypadat, ale primárně se snaží vysvětlit filozofii agilního přístupu a zaměřit se na vysvětlení, proč jednotlivé metody fungují. Součástí textu jsou praktické příklady a doporučení co dělat a čeho se naopak vyvarovat.

Kniha se zabývá agilními metodikami vývoje softwaru v různých typech společností, od malých firem až po nadnárodní korporace. V první části knihy ***‚Filozofie agilních a lean metod'*** popisuje základní směry agilního a lean světa a proč vlastně vzniká potřeba agilní transformace. Druhá část, ***Popis metod, procesů, praktik a artefaktů***, začíná slovníčkem pojmů, pokračuje detailním popisem jednotlivých rolí, praktik, artefaktů, meetingů a aktivit a končí popisem jednotlivých frameworků. Třetí část, ***Implementace, transformace, změna***, je průvodcem agilní změnou a obsahuje praktické rady pro konkrétní situace. Čtvrtá část – ***Příklady*** – obsahuje seznamy toho, co budete potřebovat změnit, a konkrétní příklady jak začít. Druhé rozšířené a upravené vydání knihu doplňuje o pátou část, ***Případové studie*** z agilních transformací v různých organizacích, a také upravuje doporučené praktiky vzhledem k současnému state-of-the-art agilního světa.

V posledních 10 letech se agilní metody staly ve světě velmi populární. Používají je velké nadnárodní korporace jako Amazon, Salesforce, Oracle či Google, najdete je na universitách, v telekomunikacích, v bankách a pojišťovnách, v armádě, aerolinkách, automobilovém průmyslu, medicínských firmách a samozřejmě i v mnoha malých či středních firmách. V České republice se dostávají do pově-

domí firem až v posledních několika letech, ale být agilní, používat Scrum či Kanban již dávno není výjimkou a agilní metody se tak stávají na českém trhu běžnou metodikou.

Kniha je primárně určena pro lidi pracující v ICT, a to jak pro softwarové vývojáře, testery, designéry, architekty a analytiky, tak i projektové a produktové manažery, vedoucí oddělení, manažery i ředitele či vlastníky takových organizací. Přestože agilní metody byly primárně definované pro zlepšení a zefektivnění vývoje softwaru, používají se v podstatě kdekoli, kde je třeba řídit projekty, organizovat týmy lidí a kde je výsledný produkt komplexní a pro jeho úspěch je vyžadována kreativita, flexibilita, schopnost reagovat na změny, např. marketingové či reklamní agentury, konzultantské společnosti, HW firmy. Máte-li na starosti nějaký projekt nebo tým lidí, určitě je tato kniha pro vás.

A kladete-li si hned v začátku otázku, co můžete spolu se zavedením agilních metod očekávat, tak vyšší flexibilitu a schopnost reagovat na změny, vyšší efektivitu, kvalitnější produkt, větší předvídatelnost termínu dodání a vyšší spokojenost zákazníka s vaším produktem či službou, a v neposlední řadě motivovaný tým, který se spolupodílí na produktu a jeho inovacích. Tedy jednoduše řečeno být úspěšnější.

Autoři vycházejí z vlastní zkušenosti s nasazováním agilních metodik v různých prostředích firem, od malých startupů až po mezinárodní korporace. Ze svých bohatých zkušeností autoři vybrali vše podstatné a srozumitelnou formou předkládají čtenáři návod, jak postupovat, od prvních krůčků až po kompletní agilní transformaci celé společnosti.

Zpětná vazba od čtenářů

Nakladatelství a vydavatelství Computer Press, které pro vás tuto knihu připravilo, stojí o zpětnou vazbu a bude na vaše podněty a dotazy reagovat. Můžete se obrátit na následující adresy:

Computer Press
Albatros Media a.s., pobočka Brno
IBC
Příkop 4
602 00 Brno

nebo

sefredaktor.pc@albatrosmedia.cz

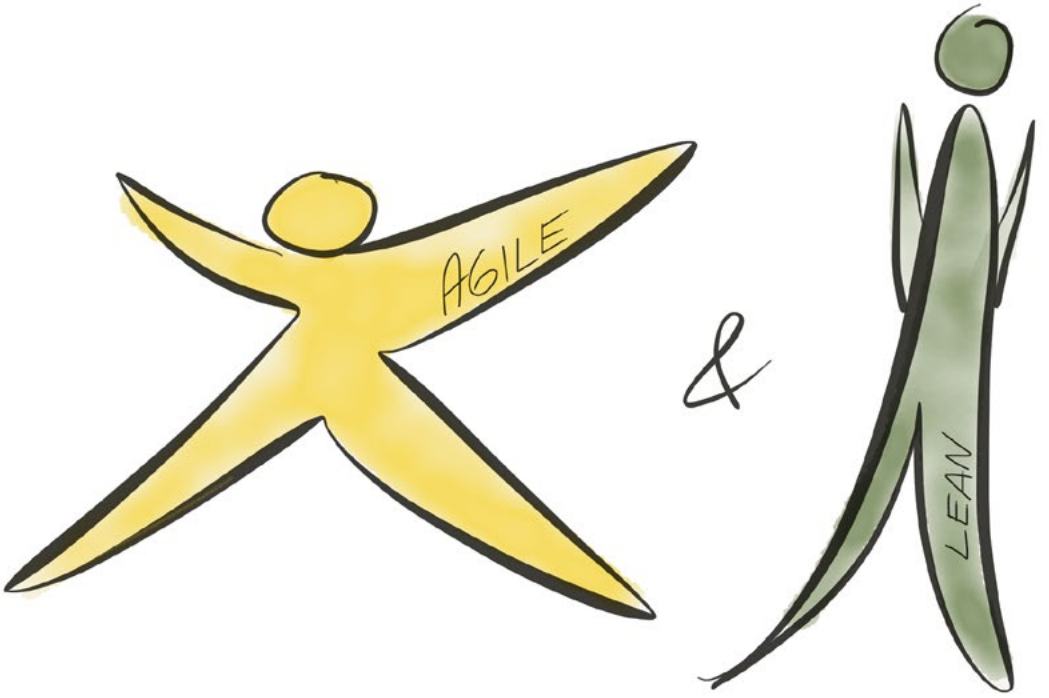
Computer Press neposkytuje rady ani jakýkoli servis pro aplikace třetích stran. Pokud budete mít dotaz k programu, obraťte se prosím na jeho tvůrce.

Errata

Přestože jsme udělali maximum pro to, abychom zajistili přesnost a správnost obsahu, chybám se úplně vyhnout nelze. Pokud v některé z našich knih najdete chybu, budeme rádi, pokud nám ji oznámíte. Ostatní uživatele tak můžete ušetřit frustrace a pomoci nám zlepšit následující vydání této knihy.

Veškerá existující errata najdete na stránce knihy na *www.albatrosmedia.cz* po klepnutí na odkaz Soubory ke stažení. (Nejsou-li žádná errata zatím k dispozici, není odkaz Soubory ke stažení dostupný.)

Část I
Filozofie agilních a lean metod


AGILE
&
LEAN

Agilní a lean metody

Co si představíte pod slovem „agilní“?

Agilní je dynamický, rychlý, interaktivní, přizpůsobivý, iterativní, zábavný, hravý, rychle reagující na změnu, … a jistě vymyslíte spoustu dalších synonym. Je to jiný způsob života, upřednostňující jiné hodnoty. Reálný výsledek před striktními procesy, změnu před předem naplánovaným. Být agilní znamená žít agilní filozofií, je to odlišná firemní kultura a nálada. Ale jestli čekáte kuchařku, jak se stát agilní, budete zklamaní. A jestli očekáváte nějaký model či framework s několika stupni agility, nebo dokonce checklist, tak vás bohužel zklameme. Nic takového nehledejte, protože to ani neexistuje, ač mnoho certifikovaných hlav tvrdí pravý opak. Žádný certifikát agility vám v pochopení nepomůže, maximálně může nastartovat proces přeměny. Ale agilní musíte být, agilně musíte myslet, agilně se chovat.

Ale nebojte se, není to zas tak neuchopitelné, jak se na první pohled zdá. Agile je o spolupráci a komunikaci a připravenosti na změnu. O tom, že zásadně děláme to, co má v danou chvíli smysl, a děláme to tak, jak nejlépe umíme. Agile sice není žádný striktní proces, ale není to ani žádný chaos. Má svá jasná pravidla. Dalo by se říct, že definuje hranice a vytyčuje menší hřiště, v jejichž rámci si týmy mohou stanovovat svá vlastní pravidla hry, tak aby se jim dobře pracovalo a byly co nejvíce produktivní, efektivní a dodali kvalitní produkt v co nejkratším čase. Takový tým je zaměřený na business value, tedy hodnotu pro zákazníka. Na to, jak optimalizovat funkcionalitu tak, aby zákazník byl maximálně spokojený a dostal za vynaložené prostředky to, co opravdu potřebuje a může používat.

Základním stavebním kamenem je – jak jinak – manifest; v tomto případě přímo agilní manifest, který shrnuje ve čtyřech bodech, co to vlastně znamená být agilní.

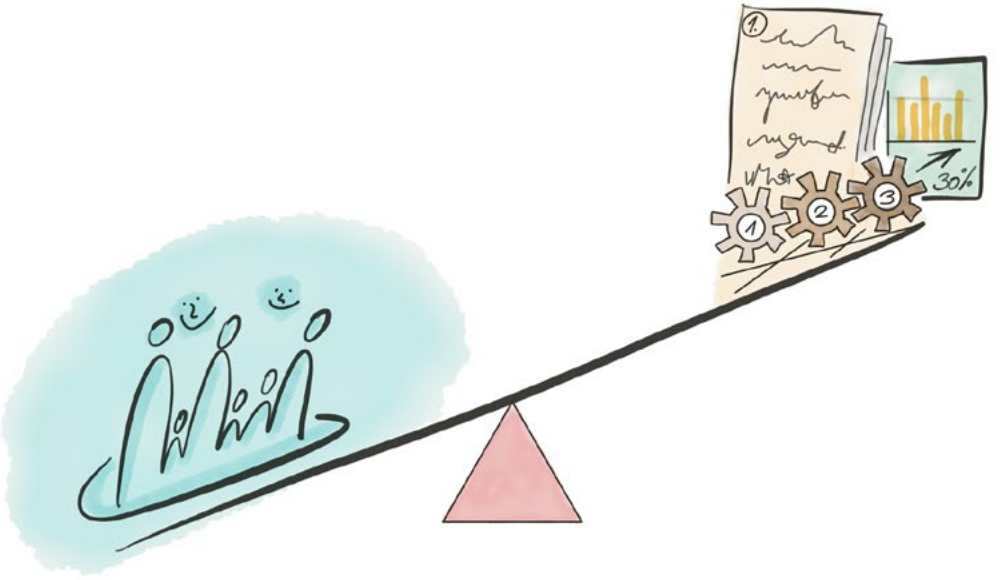
Manifest agilního vývoje softwaru

http://agilemanifesto.org

Objevujeme lepší způsoby vývoje softwaru tím, že jej tvoříme a pomáháme při jeho tvorbě ostatním. Při této práci jsme dospěli k těmto hodnotám:

- Jednotlivci a interakce před procesy a nástroji
- Fungující software před vyčerpávající dokumentací
- Spolupráce se zákazníkem před vyjednáváním o smlouvě
- Reagování na změny před dodržováním plánu

Jakkoliv jsou body napravo hodnotné, bodů nalevo si ceníme více. Pojďme se společně chvíli zamyslet nad tím, co nám vlastně manifest říká.

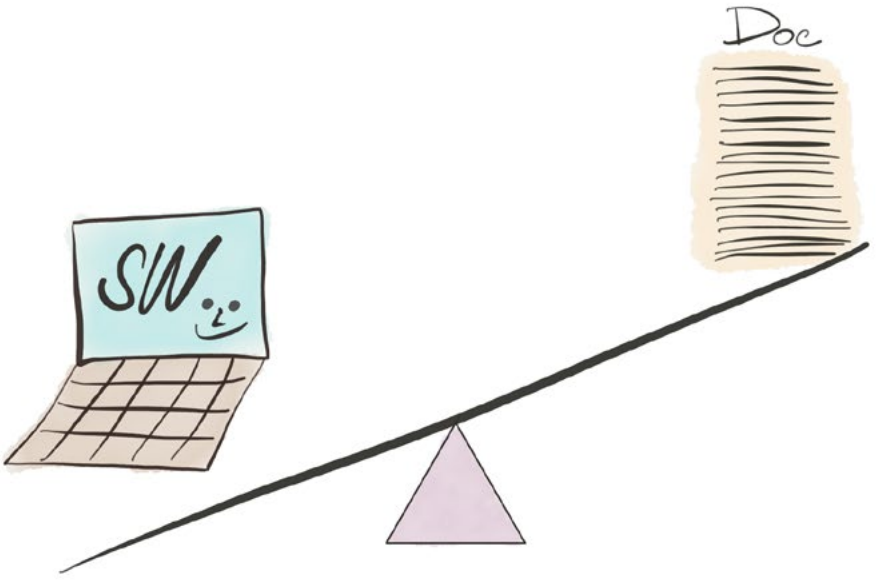


Jednotlivci a interakce před procesy a nástroji

Je známým faktem, že spolupracující týmy mají lepší výsledky než skupiny individuálně pracujících jednotlivců. Procesy a nástroje jim pomáhají dosáhnout výsledků, ale nejsou pro jejich úspěch nijak klíčové. Pojďme se podívat na dva odlišné týmy. V prvním máme 10 profesionálů, kteří si vytvořili vlastní nástroje a pracují v silně kolaborativním prostředí, ve druhém máme 10 vývojářů, kteří pracují v přesně definovaném procesním prostředí s nejlepšími nakoupenými nástroji, ale bez interakcí a spolupráce.

Co myslíte, který z těchto týmů vytvoří lepší software v kratším čase? Myslím, že se shodneme na tom, že první tým má větší šanci uspět, protože vzájemná spolupráce a komunikace mnohokrát převáží přesně definované procesy a nástroje. Ostatně podívejme se na úspěšné startupy, kde moc procesů nenajdeme, zato všichni překypují spoluprací a komunikací. No, a hlupák i s nejlepším nástrojem je zkrátka pořád hlupák.

Na druhou stranu, manifest neříká nic o tom, že procesy a dohody by neměly existovat ani že by týmy měly pracovat zcela bez nástrojů. Jen by měly mít možnost si nástroje vybrat a používat jen ty, které jim opravdu pomáhají v dosažení kvalitního výsledku.


Doc
SW

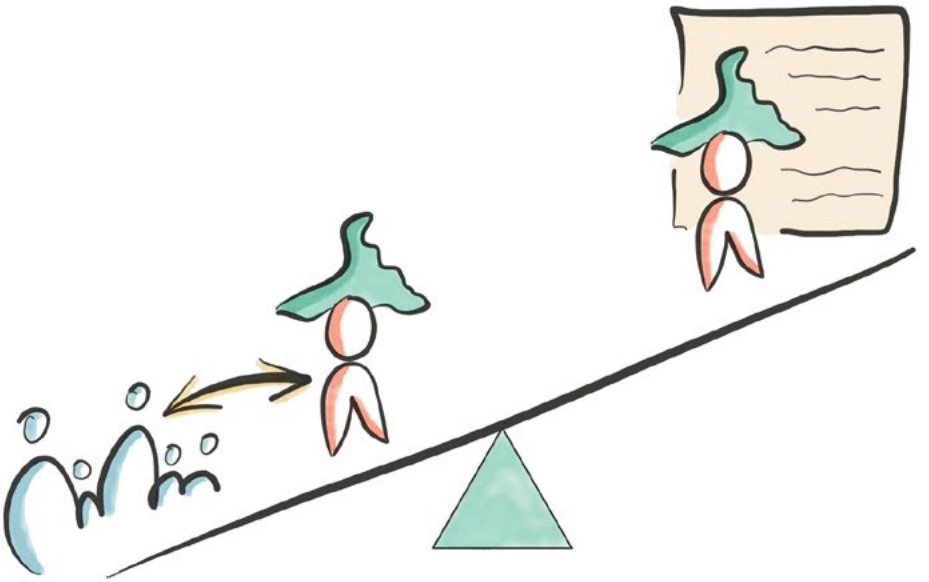
Fungující software před vyčerpávající dokumentací

Když se zeptáte zákazníků, zda by si koupili dokument popisující váš popis plánovaného produktu s architektonickou dokumentací, nebo ten samý produkt s minimální dokumentací, co si vyberou? Nebo praktičtější příklad: když si jdete koupit nový mobilní telefon, vyberete si pouze podle technické specifikace na papíře, nebo bude pro vás důležité si vyzkoušet i vlastní telefon před tím, než si ho koupíte? A když si koupíte nový televizor, přečtete si nejdříve manuál, nebo chcete vyzkoušet, jak funguje, a manuál čtete až na posledním místě? Lidé (ano, i zákazníci jsou lidé) zkrátka preferují praktické seznamování s produktem, předteoretickým.

Mnoho softwarových firem na toto zapomíná a chce ohromit zákazníka množstvím dokumentace, ale když dojde na vlastní demo, výsledek často dokumentaci odporuje a zákazník je z „funkčního" softwaru poněkud překvapený. Dokumentace **je** důležitá, ale neměla by převážit nad vlastním produktem; měla by primárně sloužit jako reference pro oblasti, které nejsou intuitivní a snadno pochopitelné. A takových oblastí by mělo být minimum.

Interní dokumentace se často píše pro budoucí týmy, aby se znalost architektury produktu neztratila. To samo o sobě není nijak špatný nápad, ale už jste někdy zkoušeli něco dohledat v deseti úrovních dokumentů, každý o 1000+ stranách? Vyčerpávající. A když už si dáte tu práci a prohledáte tu kupu slov, v devíti z deseti případů zjistíte, že daná věc v dokumentaci chybí či je v poslední verzi již zcela jinak. Interní dokumentace by měla být stručná a postihnout klíčové informace. Ostatní je obvykle efektivnější dokumentovat přímo v kódu.

Poslední případ je dokumentace funkcionalit v průběhu vývoje, kterou můžete v podstatě úplně odstranit a nahradit dobrou komunikací mezi analytiky, testery a vývojáři. Ti se pak sami domluví, co je třeba zdokumentovat pro ‚budoucí generace' vývojářů a testerů a v jaké podobě. Na závěr připomeneme, že nedoporučujeme přestat dokumentovat, ale jen dokumentaci omezit na minimum, aby poměr mezi námahou a časem, který investujete do psaní dokumentace, odpovídal hodnotě, kterou její zákazníci, tedy všichni, kteří ji kdy čtou, z takovéto dokumentace načerpají.

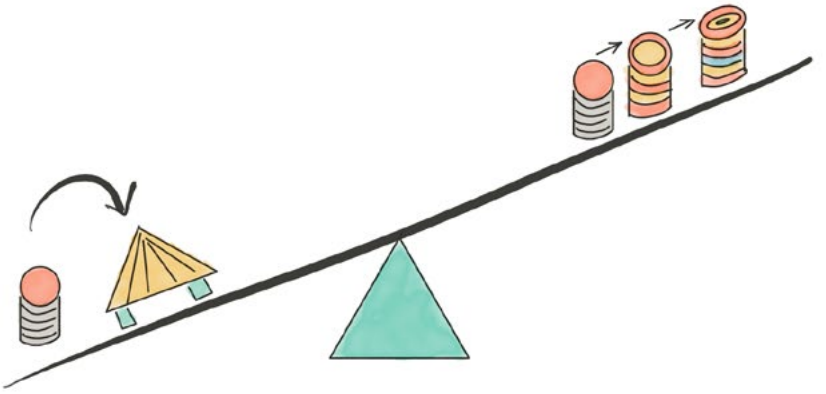


Spolupráce se zákazníkem před vyjednáváním o smlouvě

Proč vlastně vytváříme nějaký produkt? Chceme ho prodat, že? Když stavíme produkty pro zákazníky, je dobrý nápad se jich zeptat, co vlastně chtějí koupit. Jistě, zákazník mnohdy neví, co vlastně chce, a často mění názor, ale dlouhodobou spoluprací můžeme zákazníka „vychovat" a společně vytvořit spolupracující tým, který zajistí jak úspěch náš, tak úspěch zákazníka. Je lepší se se zákazníkem dohodnout a spolupracovat, než se potkávat pouze u soudu a komunikovat přes právní zástupce.

Smlouvy **jsou** důležité, ale neměly by být prostředkem nahrazujícím spolupráci a komunikaci. Ostatně není od věci se už při jejím podpisu zamyslet nad tím, co je opravdu reálné, že se při spolupráci stane – tedy že zákazník možná ne vždy ví přesně, co chce, a že se stejně bude měnit scope – a smlouvu se tak pokusit přiblížit realitě.

Spokojený zákazník, který dostal to, co potřebuje, vás doporučí dalším firmám, zatímco zákazník, který sice má to, co si objednal a ve smlouvě podepsal, ale nijak mu to v jeho misi nepomohlo, se příště podívá po jiném dodavateli. Smlouvy určitě pište, ale ani sebelepší smlouva vás neochrání před změnami, které přijdou. Obvykle to stejně bez ohledu na smlouvu skončí kompromisem, kdy po měsících hádek o chyby a change requesty jich dodavatel polovinu dodělá jako opravu na vlastní náklady a polovinu jako novou funkčnost za extra peníze od zákazníka. Vzájemné spokojenosti a dobrým vztahům to nijak nepomůže.

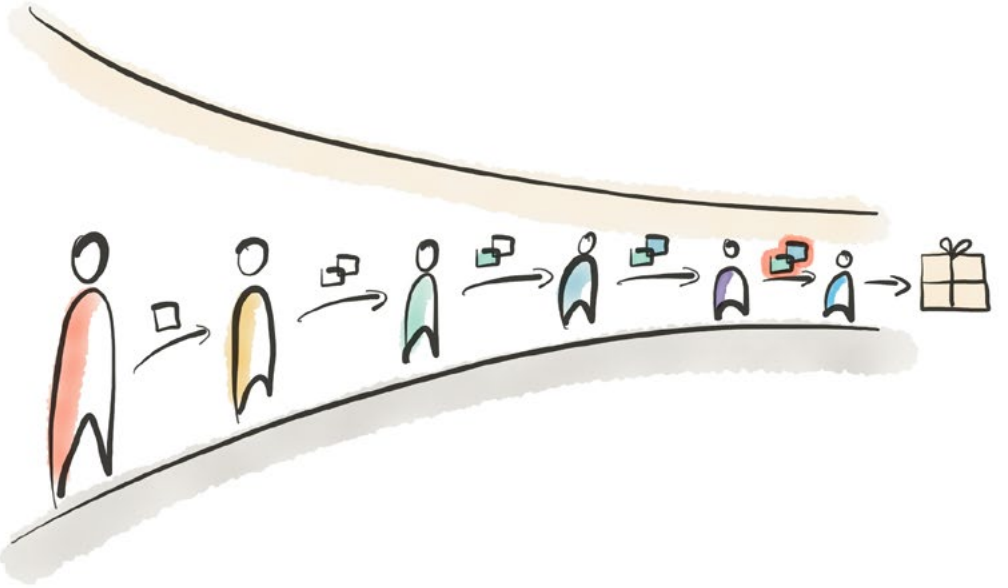


Reagování na změny před dodržováním plánu

Svět se pořád hýbe kupředu a technologie se neustále mění a spolu s nimi se mění i požadavky a problémy našich zákazníků. Ti, jako všichni ostatní, se musí přizpůsobovat trendům na trhu a konkurenci. A my, jako jejich dodavatel, je nemůžeme v těchto změnách brzdit – tedy pokud chceme dále spolupracovat. Představte si, že zákazník přijde se změnou v posledních fázích projektu a ta změna bude důležitá pro jeho přežití na trhu.

Řeknete mu, že se budete držet plánu a kontraktu a změny přijdou až po doručení první verze. A opravdu si myslíte, že další změny budou? Odvážím se tvrdit, že pokud se my dodavatelé nepřizpůsobíme, náš zákazník už si nic dalšího nekoupí, zkrátka proto, že ho konkurence převálcuje. Podobně je to i v méně vyhraněných případech, kdy zákazník prostě jen zjistí, že to, co původně tolik chtěl, vlastně vůbec neřeší jeho problémy a že by naopak potřeboval něco úplně jiného, o čem se předtím vůbec nemluvilo.

Projektové plány **jsou** důležité jako vodítko, ale neměly by řídit životy spolupracujících firem. Každé plány se mění a dogmatické dodržování původních plánů mnohdy vede k větší katastrofě než jejich postupné přizpůsobování dané situaci.



Co je to „lean“?

Lean na druhou stranu je proces převzatý z tovární výroby. Štíhlý. Dělejte věci, jen když jsou potřeba. Just in time. Často se používá i výraz systém tahu. Stejně tak jako agile i lean je spíše o přístupu než striktním procesu. Navíc agile a lean jsou si v mnohém podobné a vzájemně se prolínají. Je to takový pěkný moderní buzzword. Lean firma.

Spousty velkých firem Lean principy implementují, obvykle bez většího porozumění jejich zaměstnanci. Často to končí tím, že si udělají nějaké lístečky a jsou dostatečně lean, tedy štíhlí. Jenže na to, aby to přineslo nějaké výsledky, je stejně jako u agilních metod třeba porozumět filozofii. A ne jen slepě vykonávat nějaké rituály. O co tedy jde? Jednoduše řečeno o omezení práce na tom, co by nemuselo přinášet hodnotu, a tedy v konečném důsledku mohlo přijít nazmar.

Nejznámější lean firma je určitě Toyota. Tam vyvinuli proces řízení výroby odlišný od běžných procesů, kdy vyrábíme kdykoli a cokoli na sklad. Řídí výrobu systémem tahu, kde vyrábíme příslušný díl, až když je opravdu potřeba.

Jak takový princip použít ve firmách, kde žádné fyzické díly nevyrábíme? Tak se třeba podívejme na standardní vývojový proces – waterfall. Nejprve uděláme na sklad analýzu, pak kód a pak testy. A čekáme, že to je tak v pořádku a že všechny díly jsou kvalitní (tedy že design už se nezmění, v kódu se nenajde chyba a že zákazník to tak opravdu chce). A stejně jako ve výrobě se nám děje, že jednotlivé věci musíme předělat, opravit, zahodit. Někdy i celou krabici dílů se stejnou chybou (někdy i celou rozsáhlou funkcionalitu našeho softwaru).

Jak na to? V kostce: omezte rozdělanou práci (‘work in progress‘) a soustřeďte se na to, abyste jednotlivé požadavky dokončili co nejrychleji. Implementujte systém tahu a nezačínejte s analýzou, dokud nemáte prioritní požadavek od zákazníka. A dokud nemáte zpětnou vazbu, že předchozí požadavek byl akceptován.

Aby to bylo celé více uchopitelné, Lean Software Development je založen na následujících principech:

- **Odstraňte vše, co nepřináší hodnotu** – tedy zbavte se odpadu. Pracovat na něčem, co se ve finále vyhodí, je škoda času. Když se vám podaří tento čas investovat do věcí, co mají smysl, budete jistě efektivnější.
- **Zlepšujte se a učte se již v průběhu** – když jen slepě vykonáváte předpisy a sledujete procesy, může se stát, že stejnou chybu opakujete pořád dokola a „odpad“ se vám tedy na konci projektu nahromadí víc, než byste si přáli. Pravidelná zpětná vazba vám pomůže se soustředit jen na to, na čem opravdu záleží.

- **Rozhodujte se co nejpozději** – čím později rozhodnutí padne, tím více máte informací. Takže jsme zase zpět u myšlenky, že nemá smysl vyrábět zásoby na sklad jen proto, že zrovna máte volnou linku nebo programátory.
- **Dodávejte práci, jak nejrychleji to jde** – čím dříve něco dokončíte, tím dříve dostanete zpětnou vazbu, kterou můžete hned v další iteraci zohlednit.
- **Dejte týmu důvěru a zodpovědnost** – a budete mít mnohem motivovanější tým, než když se budete držet tradičních top-down struktur.
- **Zaměřte se na celkový výsledek** – jednotlivé chyby a selhání nejsou podstatné, jestliže se z nich poučíte. „Think big, act small, fail fast; learn rapidly" – tedy Přemýšlejte dopředu, začněte u malých věcí, ty vyhodnoťte a rychle se z nich poučte. Jen tak zajistíte, že výsledný produkt bude úspěšný. Produkt není jen výsledný software, zaměřte se na celkový dojem, který produkt vyvolává. Dbejte na kvalitu a celkovou udržitelnost systému, nevytvářejte technický dluh.

Metoda, která aplikuje na softwarový vývoj myšlenky leanu, se jmenuje Kanban a stojí někde na pomezí agilních metod a leanu. Ostatně obě metodiky staví na stejném filozofickém základu a je často těžké je oddělit. Tato kniha není primárně o lean principech, ale o agilních metodikách. Takže i když se o lean občas otřeme, do detailu se tím dále zabývat nebudeme.

Co je to Scrum?

Scrum je v současnosti jedním z nejúspěšnějších a také nejpoužívanějších frameworků, jak se stát agilními. Hodí se na komplexní prostředí, kde je těžké věci naplánovat, a tak je z pohledu businessu třeba výsledný produkt iterovat, flexibilně reagovat na změny, ale vyhnout se chaosu a strategicky ho řídit. Typicky vývoj produktu. Scrum je framework[5] velmi jednoduchý na pochopení, ale jeho aplikace vyžaduje zásadní změnu přístupu, myšlení a mindsetu. Ale o to vlastně jde.

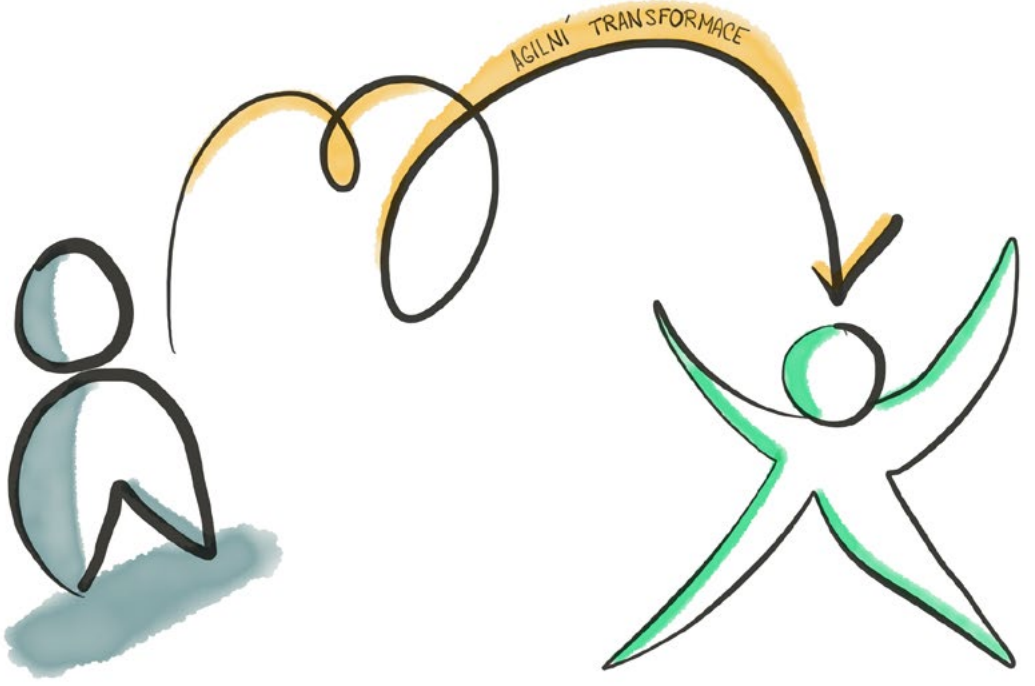
Co je to Kanban?

Kanban je proti Scrumu ještě volnější a to je také jeho hlavní slabinou. Nenutí vás totiž se nijak změnit. Kanban se hodí na prostředí, kde nic strategicky řídit nechcete a chcete jen co nejrychleji reagovat na změny. Typicky call centrum.

Co je to Scaling (škálování)?

Scaling se používá na prostředí, která jsou větší než jeden tým. Case-studies, jak na to, je mnoho (například videa Spotify), frameworků, které jsou univerzálněji popsané, je také nespočet, například Large-Scale Scrum – LeSS, který je asi nejbližší filozofii a hodnotám Scrumu. V závěru této knihy se budeme věnovat zkušenostem se scalingem, tedy škálováním Scrumu na větší celky a agilním transformacím velkých firem.


LEAN
AGILE
SCRUM
Less
DONE
KANBAN


AGILNÍ TRANSFORMACE

Agilní transformace

Agilní transformace je náročná změna. Je to změna myšlení a přístupu. Není to jen nový proces, co se někam napíše a pojede se dál. Není to jen o nějakém nástroji nebo nové roli. Je to změna kultury v celé organizaci. Agile je cesta. Nikdy nebudete hotoví. Neustále budete nacházet nové a nové metody, jak zlepšit svůj styl práce, jak najít inovativní řešení, jak se zdokonalit. Nezapomeňte, že „agile" by nikdy neměl být váš cíl. Agile je jen cesta, jak vašich strategických cílů dosáhnout.

Vývoj softwaru

Agile v prostředí vývoje softwaru začal. Čím se vlastně vývoj softwaru liší od standardní továrny? A proč klasické metody řízení projektů právě tady selhávají? Předně je vývoj softwaru komplexní a empirický problém. Nejde jen tak něco naplánovat a podle plánu vytvořit. Problém je v tom, že jakákoliv empirická činnost se velmi špatně odhaduje a ještě hůř se plánuje.

Softwarové společnosti s tím zápasí odjakživa. 70 % projektů končí pozdě, jsou mimo budget a velmi často dodají něco úplně jiného, než zákazník potřeboval. Poté následuje nekonečné dohadování o jednotlivých požadavcích, které ovšem končí vždy stejně – část zaplatí zákazník jako změnu, část dodavatel jako opravu. A o spokojeném zákazníkovi si můžeme nechat zdát.

Prvním problémem je, že zákazník často neví, co chce. Tedy vždy si myslí, že ví naprosto přesně, co chce, ale nedokáže to efektivně předat svému dodavateli. Aplikaci si neumí představit, má jen hrubou představu o tom, jak by mu měla pomoci a urychlit či jinak zlepšit práci. A typický zákazník už vůbec nerozumí počítačům, neví, proč by měl použít knihovnu X nebo technologii Y. Bohužel stále většina softwarových firem své zákazníky nutí říct dopředu přesné požadavky s představou, že oni to pak podle nich jen vyrobí, stejně jako na tovární lince.

Typicky požadavky rozmyslí tým analytiků, popíše, navrhne technologii, architekturu, design, cosi zdokumentuje a předá vývojářům. Ti interpretují specifikaci podle svého a něco nakódují. A když zbude čas, hodí to ještě testerům na otestování. Na závěr to předáme zákazníkovi a o problém je postaráno; s překvapením sledujeme upřímné zděšení zákazníka následované slovy, že očekával něco jiného. Že pracuje jinak, než jsme si mysleli, a že naše geniální funkcionality ani nepochopil. A hlavně – vůbec není spokojený. Přitom my jsme napsali přesně to, co si objednal. Výsledek? V mnoha firmách to skončí tak, že „příště si musíme dát pozor a sepsat ještě detailnější specifikaci, aby se nám to nestalo". A bludný kruh se uzavírá … anebo ne?


PROČ NĚCO MĚNIT?
PŘEDVÍDATELNOST
ZÁBAVA
FLEXIBILITA
KVALITA
EFEKTIVITA

Proč vůbec něco měnit

Hned na začátku vás musíme upozornit, že vás čeká spousta nelehké práce. Totiž zavádění agilních metodik, to není jen o změně procesu nebo kolonek na formulářích. Iniciátorem celé změny je obvykle palčivý a zdánlivě neřešitelný problém, který vás pálí natolik, že se vám přechod na agilní metody vyplatí. Každá změna je těžká a změna kultury rozhodně nejtěžší.

Čeká vás změna firmy z hierarchicky řízené na firmu orientovanou na problém. Budete chtít dosáhnout něčeho, čemu se říká samoorganizovaný tým. Najednou vaše práce nebude založená na práci jednotlivců řízených odněkud shora, ale na spolupráci lidí v týmu, na jejich schopnosti se domluvit, rozhodnout a nést za svá rozhodnutí odpovědnost. Pro některé firmy to zní jako utopie, pro jiné je to blízké tomu, jak již dnes fungují. V obou případech se ale změní hodně, a to nejen u vývojářů a testerů.

Jaké jsou nejčastější důvody pro přechod na agilní metody?

Flexibilita – do nedávna to šlo. Dělali jste releasy jednou za 6 až 12 měsíců, zákazníci na to byli zvyklí a firma také. Když chtěli nějakou změnu, protáhli jste ji vaší SW výrobní linkou a při troše štěstí zákazník svou změnu dostal už za několik měsíců. Ale doba se změnila a najednou všichni chtějí všechno hned. Chtějí dostávat výsledky po malých kouscích, ideálně ihned, jak jsou hotové. Bohužel toto v současném procesu není možné. Analytici musí požadavek popsat, vývojáři nakódovat, testeři otestovat a to není jen tak. Navíc nemůžete releasovat kdykoli. Tím byste ohrozili plán releasu a celá snaha by skončila chaosem.

Efektivita – máte pocit, že by toho šlo stihnout víc, kdybychom šli všichni za jedním cílem? Studie ukazují, že spolupráce více lidí je výrazně efektivnější než práce jednotlivců. Co se frameworků týče, máte v podstatě dvě možnosti. Zkusit párové programování, kde máte na všechny činnosti dva lidi, tedy i dva SW vývojáře u jednoho počítače. Zkušenosti ukazují, že tato metodika je efektivnější, kvalitnější a rychlejší, než když každý z nich pracuje samostatně. Anebo postavit spolupracující tým, jak doporučuje Scrum proces, kdy jednotliví členové spolupracují na jednom výsledku, pomáhají si a sami se organizují. Chvíli trvá, než si tým na sebe zvykne, ale funguje to skvěle.

Ještě na jeden aspekt se můžete zaměřit. Podívejte se na to, jak řídíte funkcionalitu. Kdo je zodpovědný za produkt a kdo za konkrétní funkčnost. Dělejte jenom na tom, co opravdu je potřeba. Na tom, co zákazník potřebuje a použije. Na tom, co mu přinese v daném čase co nejvyšší hodnotu. Systémy jsou plné funkcionalit „co kdyby“. Budete-li dobře řídit funkcionalitu, a i v tom vám agilní metody pomůžou, můžete ušetřit až 80 % času.

Předvídatelnost – jestli vaše projekty končí včas a bez přesčasů před koncem relasu, asi vám váš proces funguje dobře. Pokud to tak není, agilní metody vám mohou pomoci i s tímto aspektem. Tyto metody zavádějí zcela odlišný způsob odhadování. Odhadují v relativních jednotkách a do odhadování zapojují celý tým. Zprvu to zní hodně zvláštně, ale přesnost vašich odhadů se časem výrazně zlepší. Rozdělit projekt na malé kousky je druhou důležitou metodou, jak zlepšit předvídatelnost. Na krátkém úseku se méně projevuje tzv. studentský syndrom a effort, který tým investuje, je stabilnější, bez výkyvů a stresu na konci.

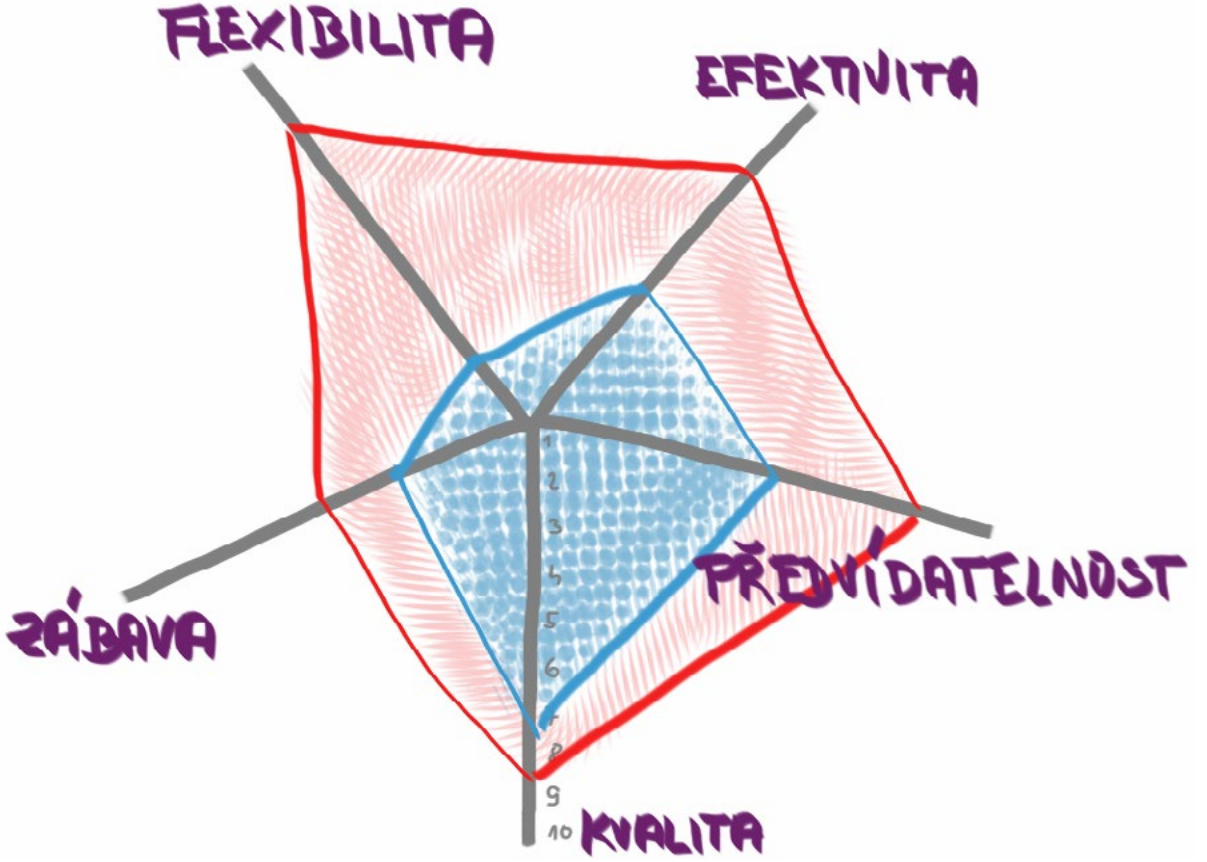
Kvalita – tady cílíme na dvě oblasti. Za prvé zapojíme do produktu zákazníka. Zeptáme se ho, co chce, proč, a taky kdo a jak to bude používat. Ukazujeme mu výsledky po malých kouskách a tím řídíme jeho očekávání. Nestává se pak, že by zákazník produkt odmítl jako nepoužitelný a trval na jeho přepsání. Na druhé straně tím, že uděláte celý tým zodpovědný za kvalitu výsledku, se sníží počet chyb a vzroste dlouhodobá udržitelnost kódu. Tým se sám stará o to, že jim neroste technický dluh. Pro oba směry zavádějí agilní metody spoustu praktik.

Zábava – a v neposlední řadě, práce bude zase zábava. I jednotliví členové budou vědět, co a proč píšou. Budou chápat smysl produktu a rozumět zákazníkovi. Navíc spolupráce s ostatními nakonec také přináší zábavu. A motivovaný člen týmu je jistě efektivnější než znuděný vývojář sedící sám za svým počítačem.

*Než se pustíte do nasazování agilních metod, udělejte si jasno v očekáváních, zejména týkajících se **flexibility**, **efektivity**, **předvídatelnosti**, **kvality** a **zábavy**.*

Jak jsme již říkali, nic není zadarmo. Jestli jste v některé kategorii našli dostatečně velký potenciál, pusťte se dál do čtení. Jestli ne, možná, že vám současné projekty fungují dobře a na změnu ještě nenastal čas. Nasazovat agilní metody jen proto, že je to něco nového, nemá smysl. Je to náročný a trnitý proces, na jehož konci vás musí čekat dostatečně velká odměna. Jinak to vzdáte ještě dřív, než se změny stihnou projevit.

Zkusit si to můžete na jednoduchém cvičení. Výše zmíněné důvody pro změnu si nakreslete do grafu a ohodnoťte jednotlivé kategorie na stupnici 1–10, kde 1 je špatné, 10 je super. Hodnotíte váš tým, projekt, produkt, firmu. Ideální je tzv. pavučinu nakreslit ve dvou barvách. První odpovídá realitě, tedy současnému stavu, a druhá odráží vaše očekávání za, řekněme, rok. Oblasti, ve kterých je největší rozdíl hodnot, indikují důvody pro změnu. Jste-li naopak se vším spokojeni, pak nejlepší strategií bude zůstat u současných procesů a nic neměnit.


FLEXIBILITA
EFEKTIVITA
ZÁBAVA
PŘEDVÍDATELNOST
KVALITA
1
2
3
4
5
6
7
8
9
10