Agilní metodiky a vývojové procesy

Co je agilní vývoj

- Primárně iterativní přístup
  - Například sprinty
- Rychlá a pružná reakce na trh
  - Požadavky na změny
  - Opravy chyb
  - Využití nových technologií
- Agilní vývoj se hodí tam kde se předpokládá dlouhodobý rozvoj

Agilní manifest

- **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,

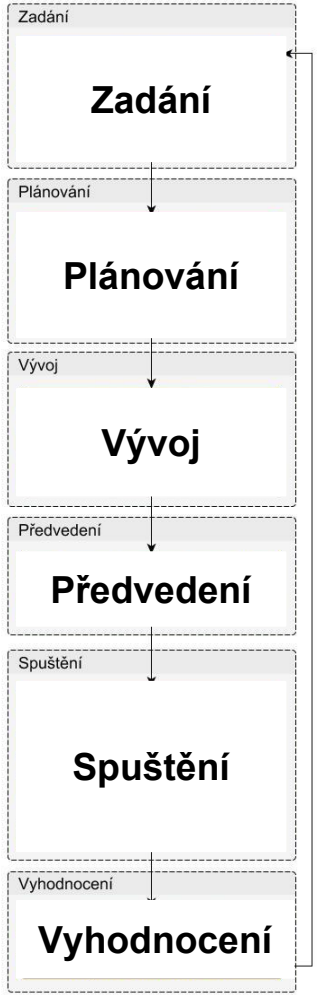
bodů nalevo si ceníme více.

Role v agilním týmu

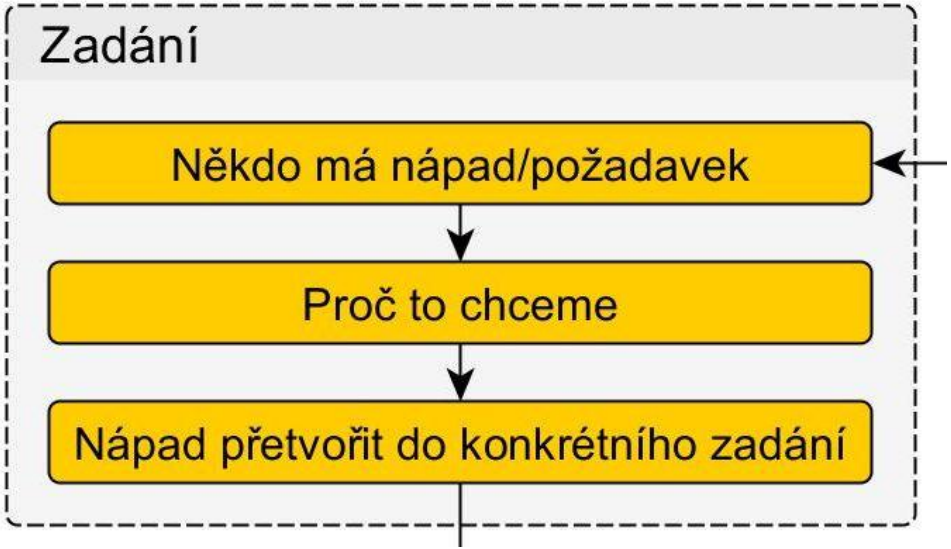
- Product owner = Produkťák
  - Sbírá podklady od ostatních na vylepšení produktu nebo vymýšlí vlastní
  - Připravuje zadání
  - Vyhodnocuje business cíle
- Scrum master
  - Dohlíží na procesy
  - Ochraňuje tým před vnějším světem
  - Tým by se o sebe měl ideálně starat sám
    - Po nějaké době už by Scrum master neměl být potřeba

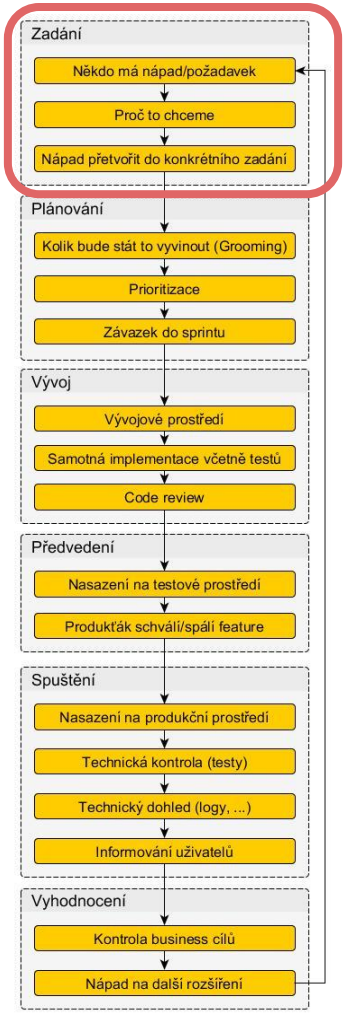
Životní cyklus feature - přehled

- Zadání
- Plánování
- Vývoj
- Předání
- Spuštění
- Vyhodnocení



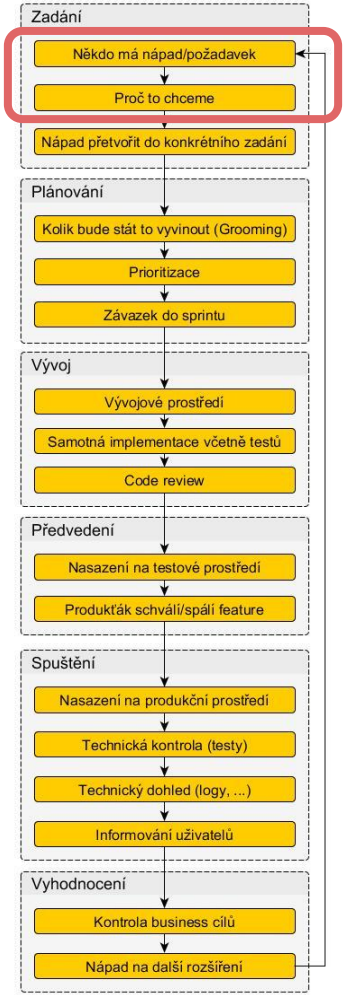
Zadání


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání
Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu
Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review
Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature
Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů
Vyhodnocení
Kontrola business cílů
Nápad na další rozšíření

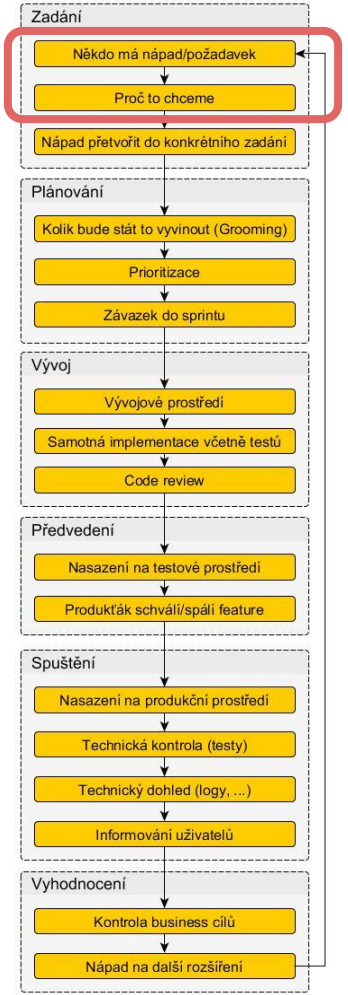
Vždy se ptejte proč danou feature děláme

- Nejen ve Scrum metodice se doporučuje tato forma zadání (feature, story, …)
  - **KDO** - pro koho story děláme
  - **CO** - co je obsahem story
  - **PROČ** - ujasnění, proč story dělám
    - Nestane se, že vytváříme něco jen tak
    - Vývojářům to pak pomůže s pochopením problému a mohou přijít s vlastním návrhem na vylepšení
- Příklad
  - Uživatel chce mít možnost zobrazit profil jiného uživatele, aby zjistil jeho kontaktní údaje



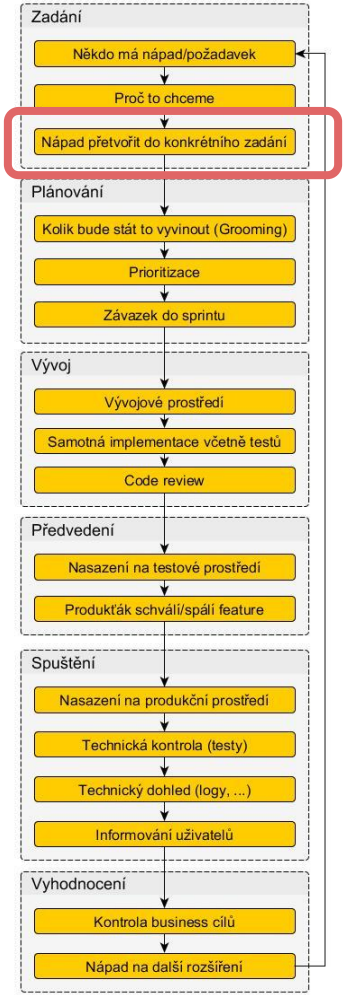
Zákazníka zajímá viditelná změna produktu

- Nová feature by měla být prezentovatelná
- Velká zadání nerozdělujte po vrstvách (DB, design, back-end, front-end, …)
- Velká zadání rozdělte po uživatelských vlastnostech



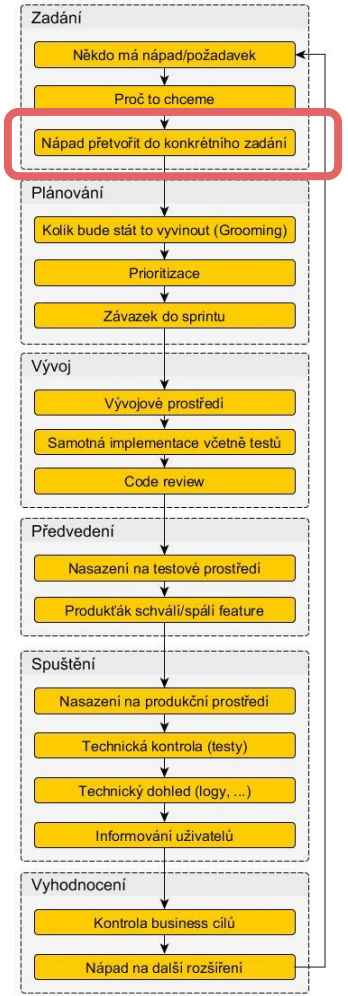
Vývojáři nedokáží odhadovat celý projekt

- A menší zadání se vývojářům i lépe odhadují
- Více menších úkolů se dobře doplňuje s iterativním přístupem
  - Malé zadání se rychle odhadne, vyvine, otestuje
  - Lehce na něj navážeme s dalším úkolem



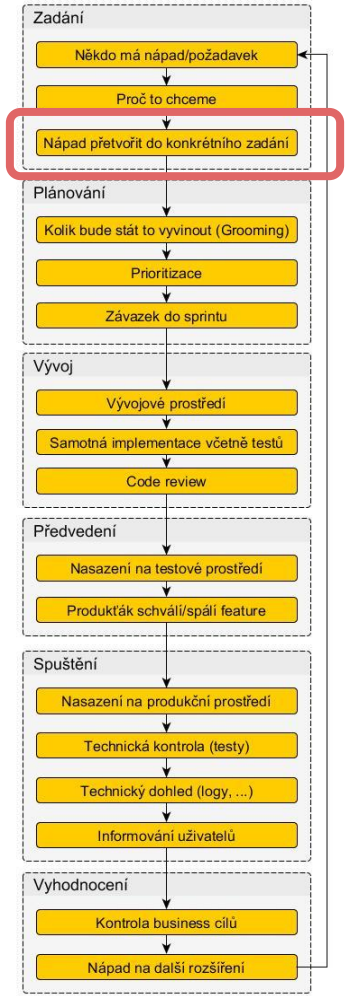
Zadání nemusí být zbytečně podrobné

- Pokud správně rozdělíme zadání na malé celky, není nutné psát rozsáhlá zadání
- Díky **PROČ**, zná motivaci úkolu i vývojář a v drobných nejasnostech dokáže improvizovat
- Pokud něco není jasné, je nejlepší se zeptat
  - Ideálně osobně, pokud je tu ta možnost

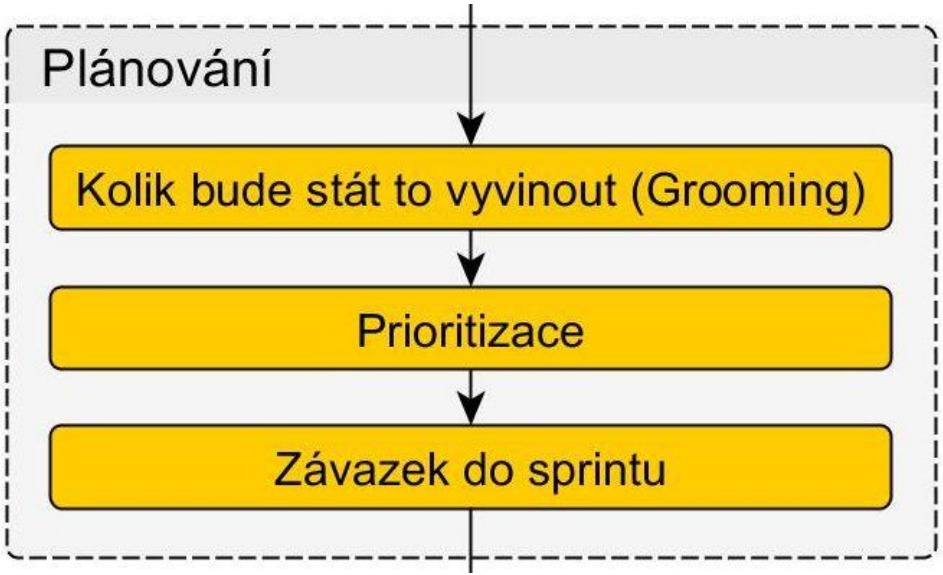


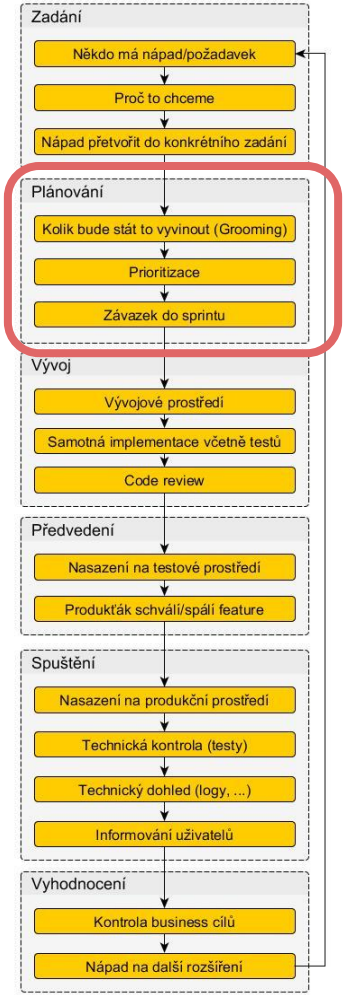
Nedělat featury dokonalé, ale tak aby se na nich dalo stavět

- Nemusíme tvořit dokonalé podrobné zadání
- Lepší je zkusit udělat prototyp a postupně jej vylepšovat
- Řešení ale musí být rozšiřitelné
  - Prototyp neznamená špatnou implementaci, ale produktově minimalizované zadání



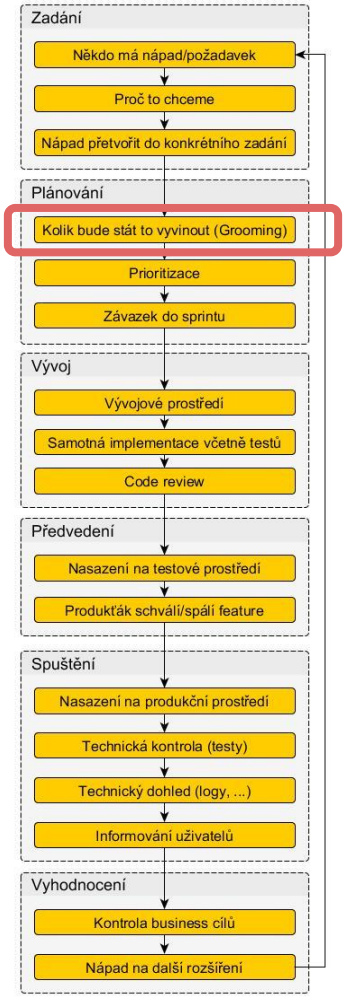
Plánování


Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání
Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu
Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review
Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature
Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů
Vyhodnocení
Kontrola business cílů
Nápad na další rozšíření

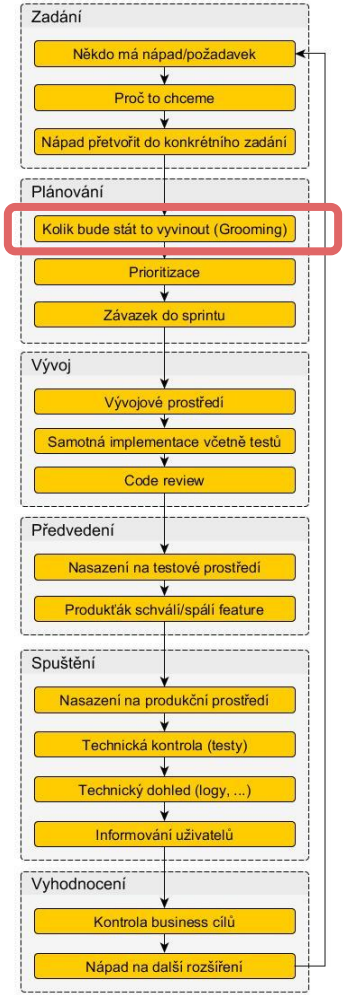
Feature hodnotíme pomocí bodů určujících složitost a ne čas

- Časový odhad je velmi nepřesný a každý jej má jiný
  - Ale na tým je potřeba se dívat jako na celek
- Snažíme se úkoly ohodnotit vzájemně mezi sebou
- Až na základě zkušenosti dokážeme predikovat budoucí rychlost týmu
  - Veškeré odhady vychází z průměrů



Výhody hodnocení pomocí bodů

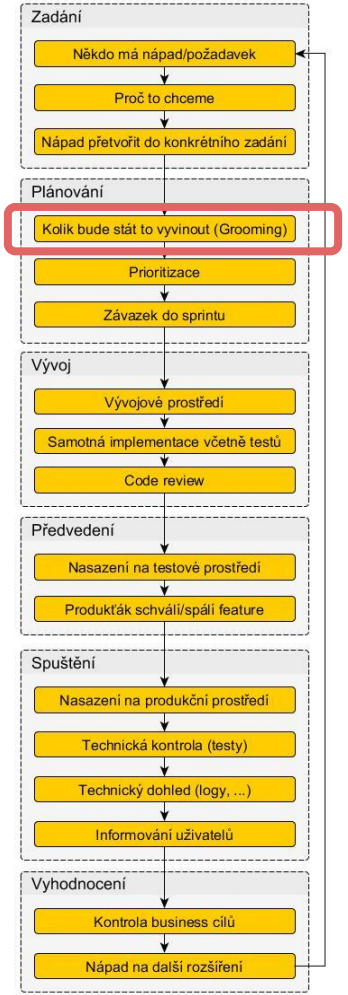
- Stabilita
  - Není nutné odhady časem dělat znovu
- Rychlost
  - Po zaškolení jsou odhady velmi rychlé
- Nezávislé na kvalitě programátora
  - V porovnání s ostatními úkoly bude každý hodnotit stejně
- Plánování do budoucna
  - Na základě minulosti, zle odhadnou rychlost v budoucnu
- Není nutné řešit agendu
  - Automaticky zahrnuje i průběžné schůzky atd.



Hodnocení featur provádí tým společně

- Na schůzce “Grooming” přednese produkťák zadání a vývojáři jej ohodnotí
- Je zde prostor na doplňující otázky a vyjasnění zadání
- Hodnocení by mělo být rychlé
  - Nemusí být naprosto přesné
  - Díky zkušenostem dokáže tým hodnotit úkoly velmi rychle

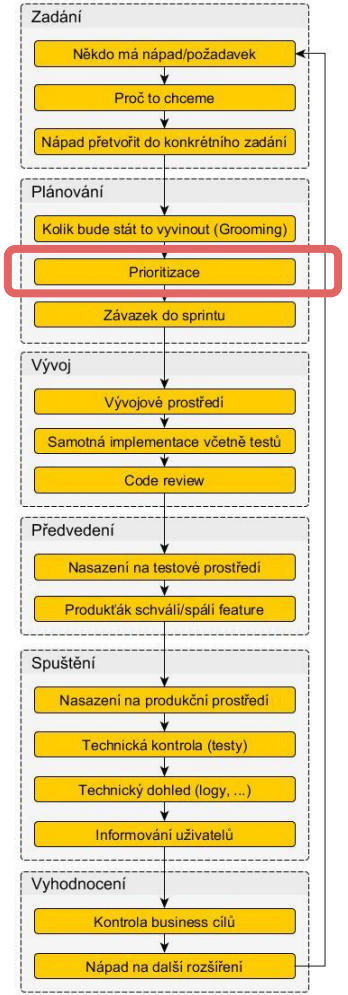
Nejdůležitější schůzka vzhledem k plánování



Prioritu určujeme podle

- Měli bychom znát business přínos
  - Výdělek
  - Úspora nákladů
  - Legislativní podmínky
  - Závazky vůči obchodním partnerům (deadline)
  - …
- Známe složitost

- Lze určit poměr CENA / VÝKON



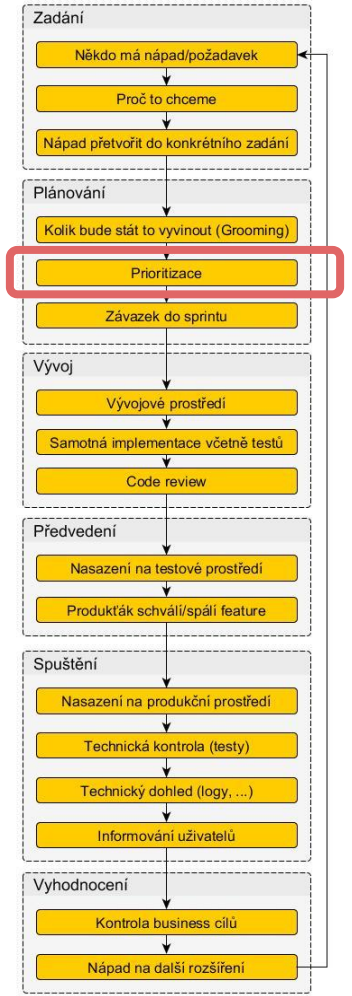
Analyzuje a plánuje se jen to nejnutnější

- To neznamená, že analyzujeme nedostatečně
- Než se něco vyvine, je to finálně dostatečně analyzováno
  - Ale je zbytečné analyzovat příliš vzdálenou budoucnost
- Nestane se, že bychom analyzovali zbytečně

podrobná analýza
- priorita úkolů
- odhad náročnosti
- přesné zadání
- dílčí podúkoly

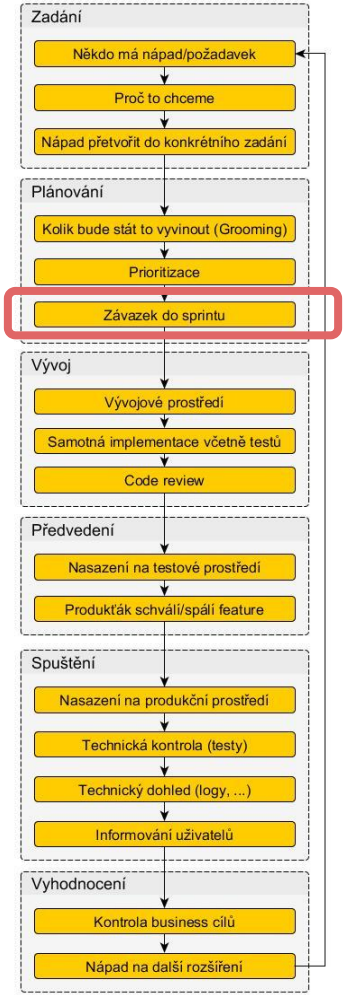
střední analýza
- priorita úkolů
- odhad náročnosti

zběžná analýza
- priorita úkolů

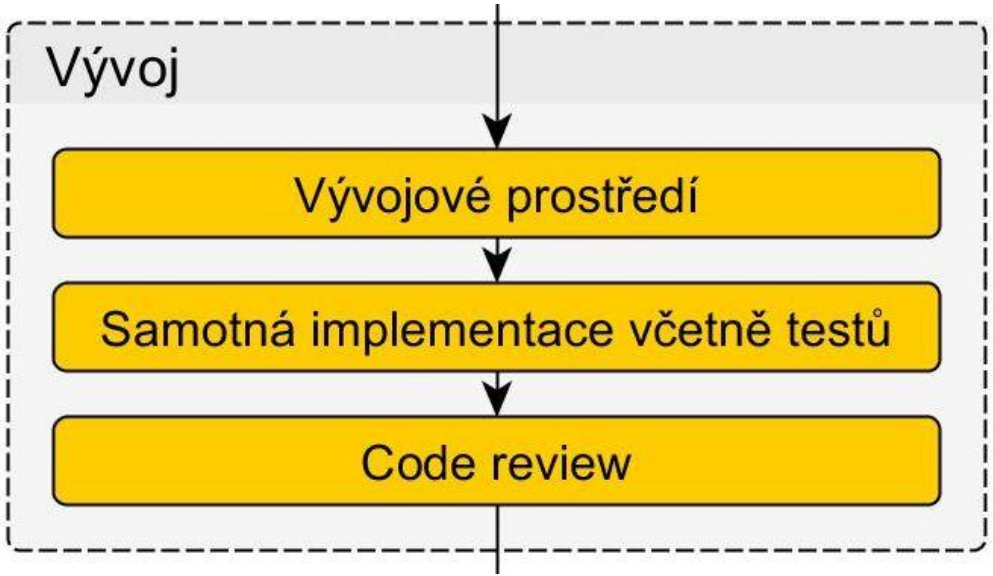


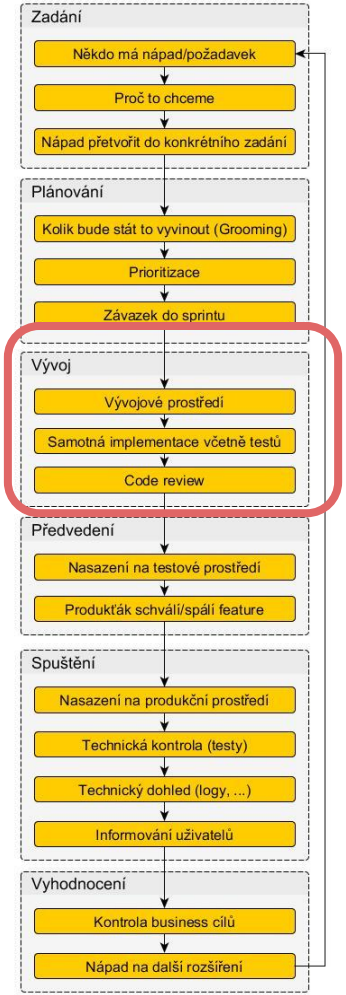
Díky zkušenosti z minulých sprintů dokážeme rychle odhadnout práci na další

- Známe hodnotu úkolů, které máme implementovat
- Známe průměrný počet bodů, které stihneme za sprint
- Ale je to jen odhad
  - Nezle nutit vývojáře k něčemu, co tvrdí, že nestihnou



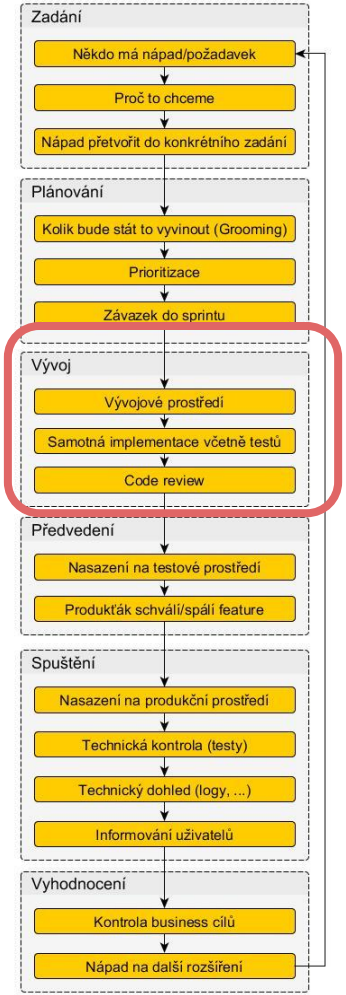
Vývoj


Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání
Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu
Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review
Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature
Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů
Vyhodnocení
Kontrola business cílů
Nápad na další rozšíření

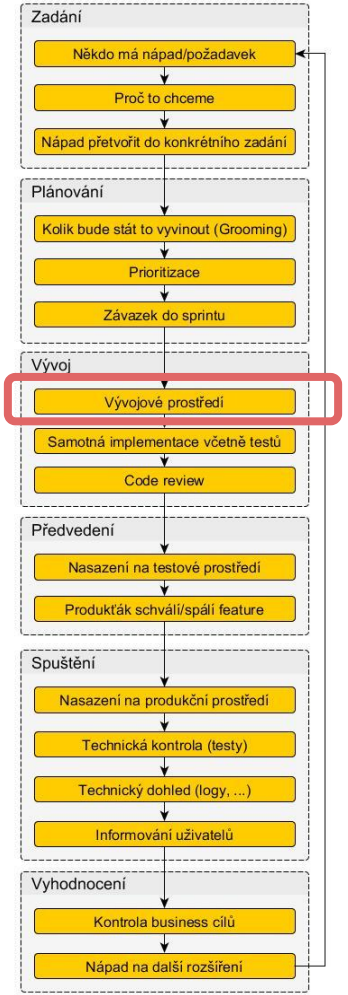
Agilní vývoj

- Sledujeme stejné mechanismy jako při přípravě zadání
- Postupujeme iterativně, po menších logických celcích, které ale 100% fungují
- Opět upřednostňujeme vývoj po funkčních celcích - nikoliv po technologiích



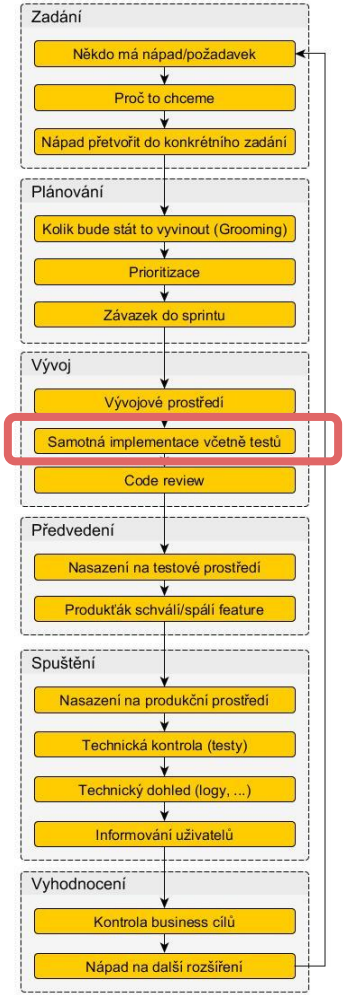
Infrastruktura

- Lokální vývoj
  - Jak dlouho trvá zaškolení nováčka?
  - Mohu si postavit můj soukromý funkční systém?
  - Kolik času se stráví odhalováním chyb v konfiguraci?
- Sdílené prostředky
  - Odkud vezmu testovací data?
- Virtualizace
  - lokální
  - cloud



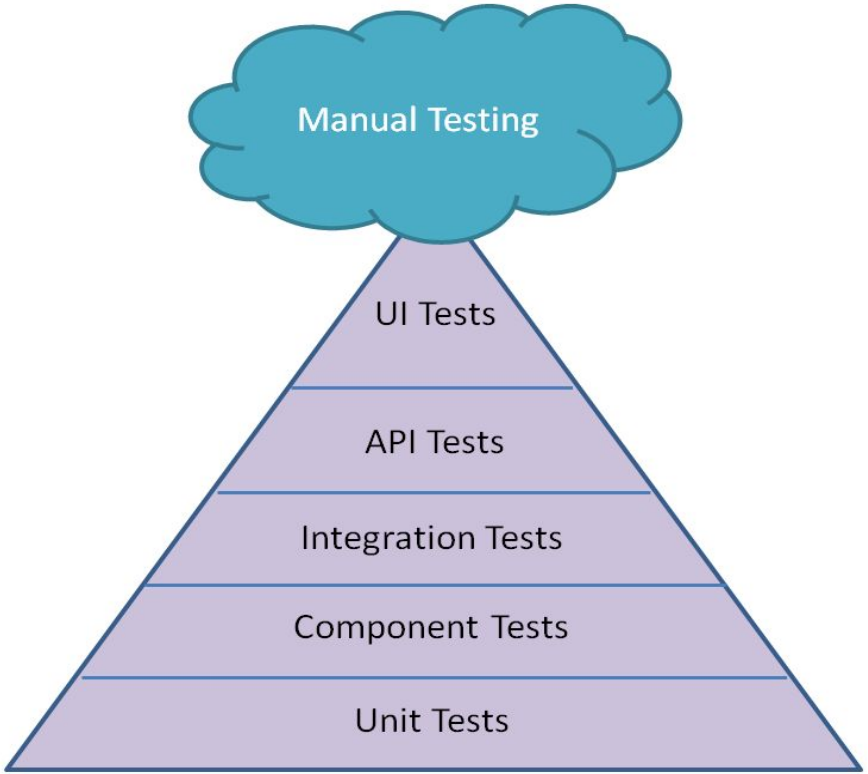
Automatické testy

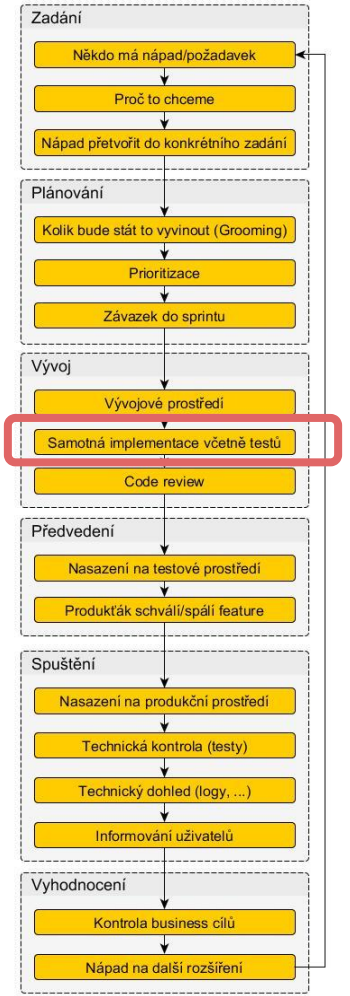
- Psaní automatických testů není zbytečné zdržování vývoje, ale přímá reakce na požadavky produktu
- Optimalizace ceny změny produktu
- Kolik nás stojí zkontrolovat, že náš produkt funguje?
- Jak vlastně zkontrolujeme že náš produkt funguje?



Automatické testy

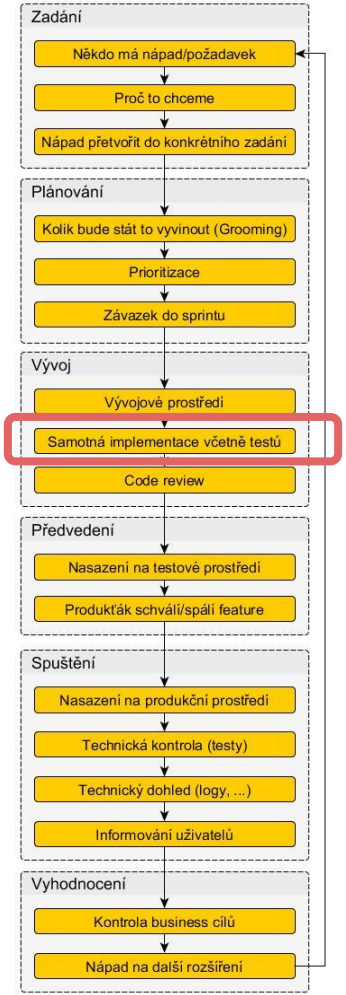
- unit
- integrační
- akceptační
- explorativní





Refactoring

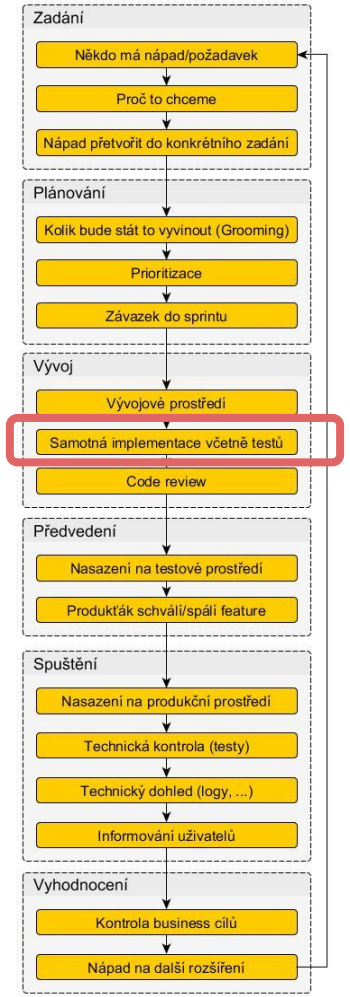
- úprava kódu beze změny funcionality
- perfektní kód bez testů přirozeně zahnívá
- špatný kód s testy se přirozeně vylepšuje



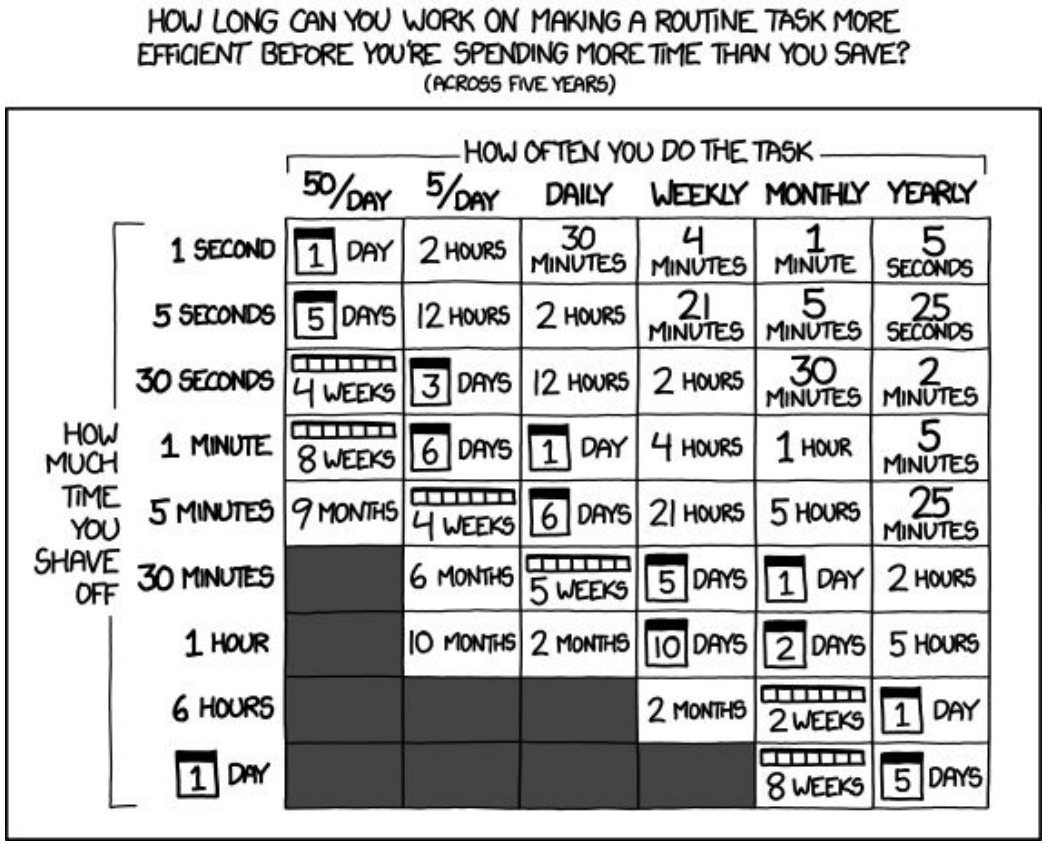
Automatizace opakujících se úkolů

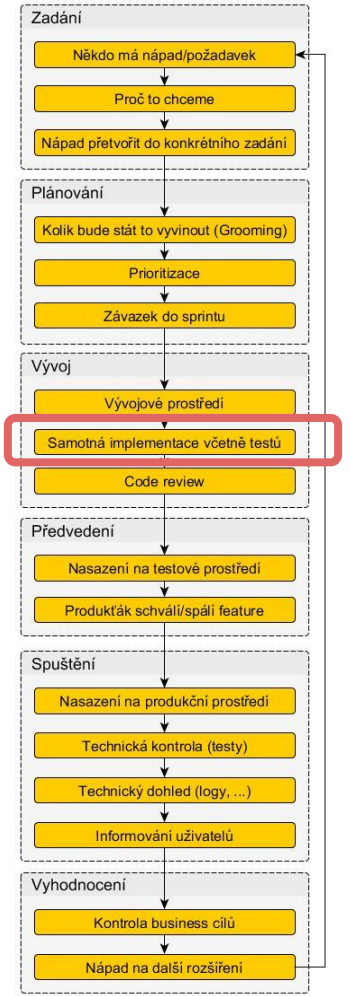
Kvalitní automatizace nám zlevní režii vývoje.

- Vývojové prostředí
- Spouštění testů
  - Continuous integration
- Příprava release
  - Continuous deployment (delivery)



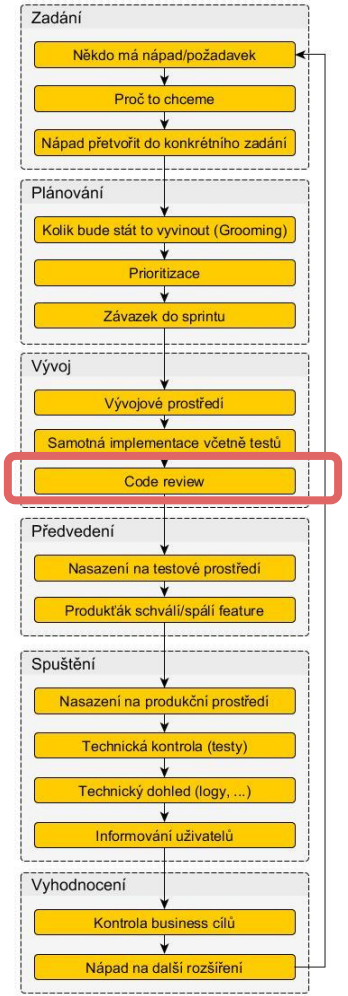
Automatizace opakujících se úkolů





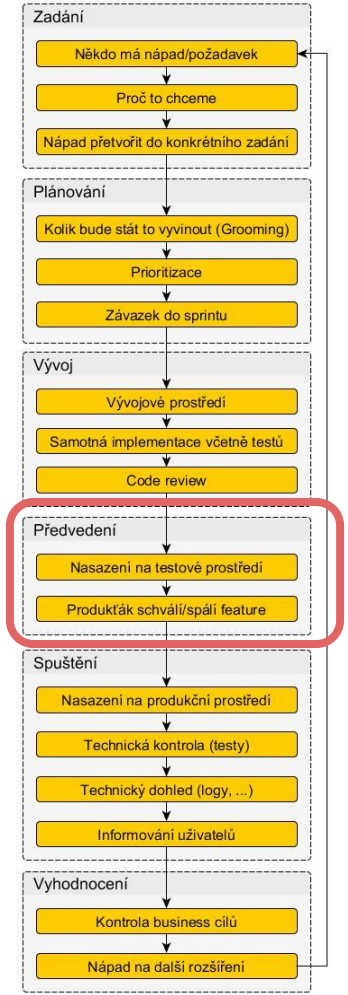
Týmová koordinace

- Verzovací systém
- Coding standard
- Code review
- Extrémní programování
  - Párové programování
  - Test driven development



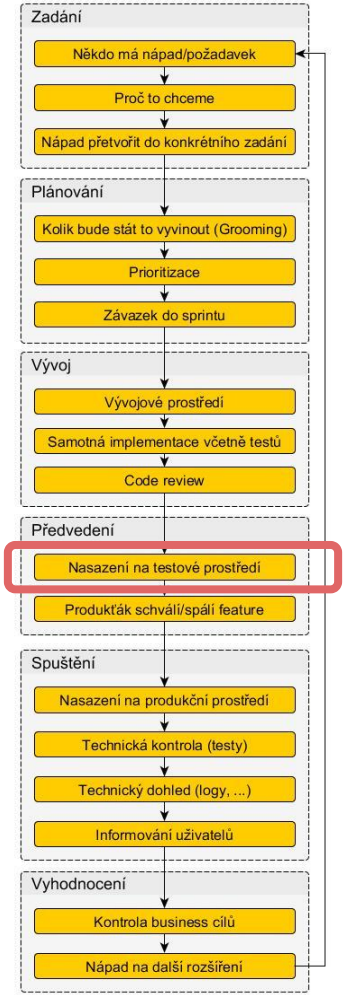
Předvedení


Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání
Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu
Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review
Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature
Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů
Vyhodnocení
Kontrola business cílů
Nápad na další rozšíření

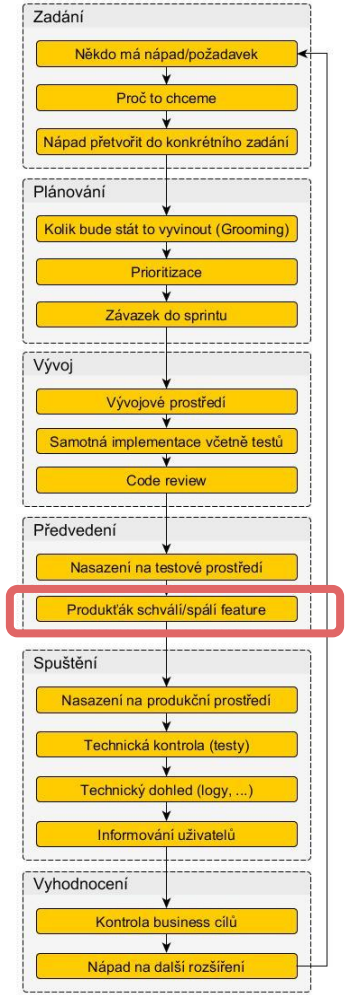
Testovací prostředí

- Zvláštní prostředí na hraní pro produktového managera
- Mělo by vždy fungovat - tedy hraje si tam produkťák, nikoliv vývojář
- Díky automatizaci můžeme mít testovacích prostředí víc (klidně jedno pro každou novou feature)

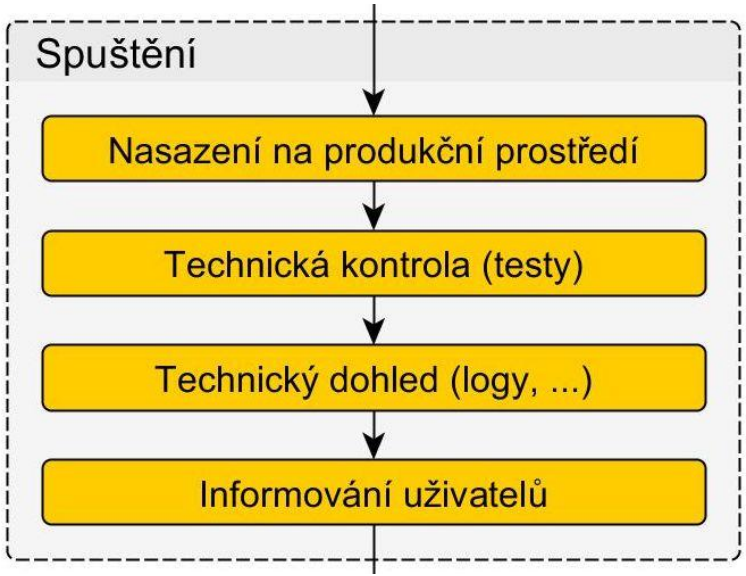


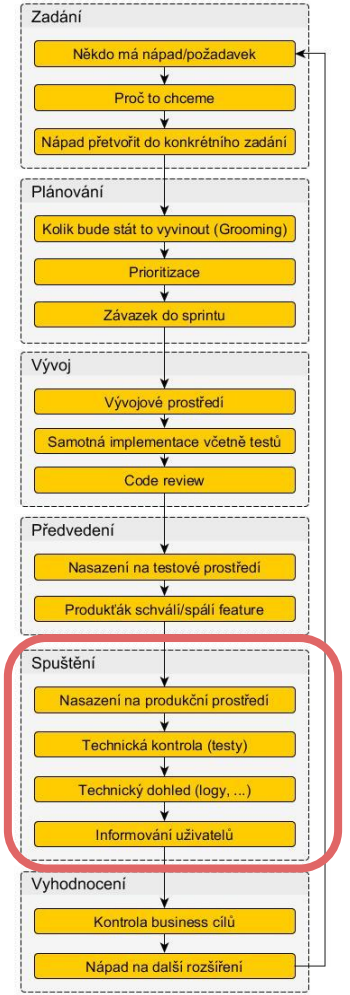
Důležitá je každodenní komunikace

- Produktově daný úkol schvaluje (tzv. spálí) produkťák
- Nejlépe na Testovacím prostředí
- Je ale vhodné, aby vývojáři s produkťákem průběžně komunikovali už při vývoji
  - Průběžně mu ukazovali co mají (třeba i na svém stroji)
  - Ptali se ihned na nejasnosti



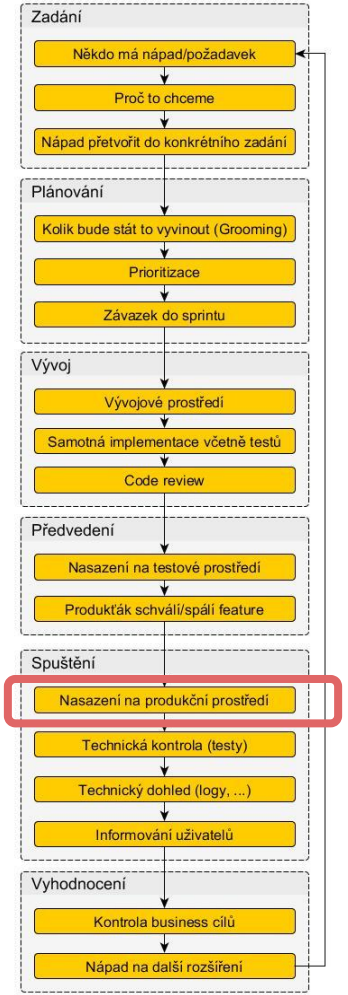
Spuštění


Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů


Zadání
Někdo má nápad/požadavek
Proč to chceme
Nápad přetvořit do konkrétního zadání
Plánování
Kolik bude stát to vyvinout (Grooming)
Prioritizace
Závazek do sprintu
Vývoj
Vývojové prostředí
Samotná implementace včetně testů
Code review
Předvedení
Nasazení na testové prostředí
Produkťák schválí/spálí feature
Spuštění
Nasazení na produkční prostředí
Technická kontrola (testy)
Technický dohled (logy, ...)
Informování uživatelů
Vyhodnocení
Kontrola business cílů
Nápad na další rozšíření

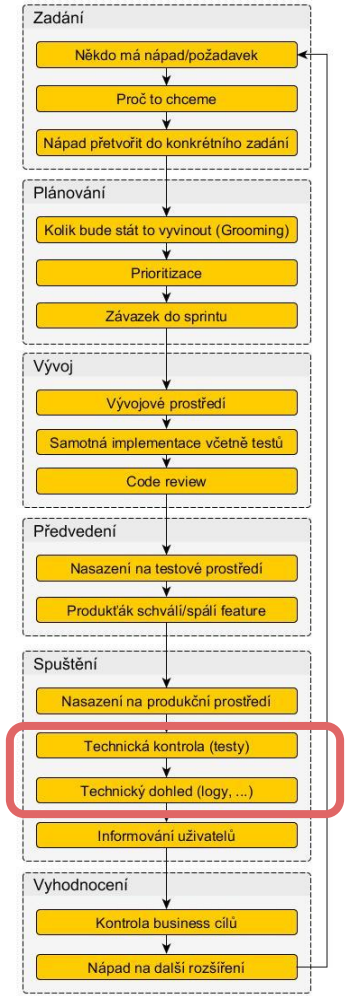
Nasazení do produkce

- Vše by mělo být vyzkoušeno na předchozích prostředí (zejména na testovacím)
- Nemůžeme chybu úplně vyloučit, tedy je třeba mít připraven plán co dělat v případě chyby
  - rollback
  - postupné nasazování
  - nasazování mimo špičku



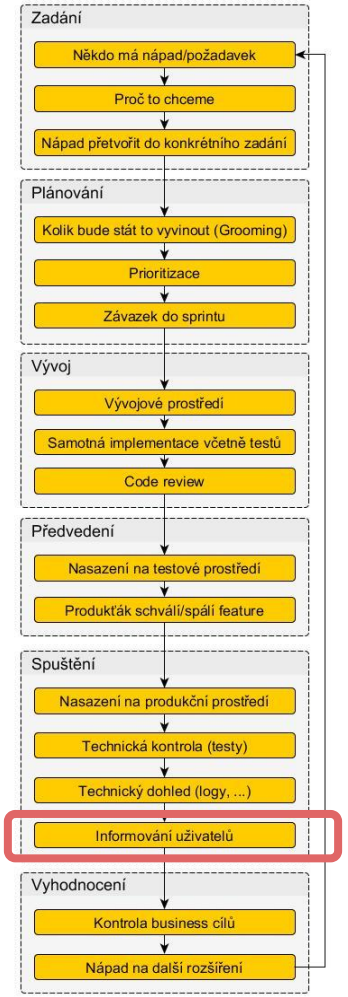
Kontrola spuštěné produkční aplikace

- Sledování logů
- Sledování výkonnostních charakteristik
- Využití nedestruktivních akceptačních testů

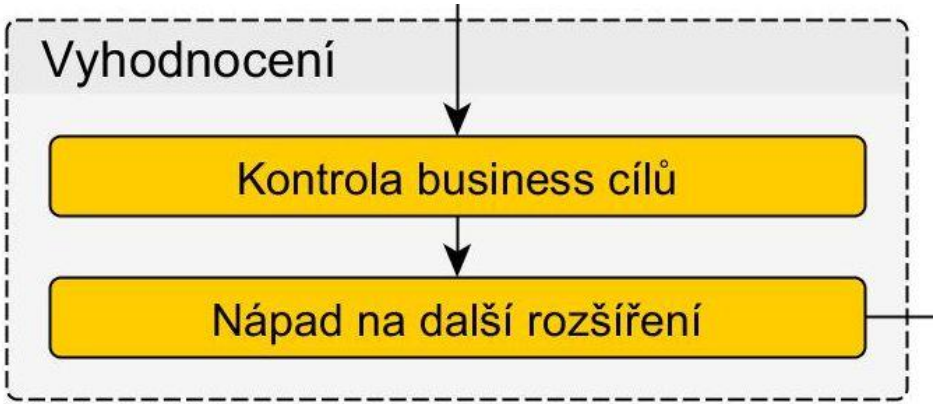


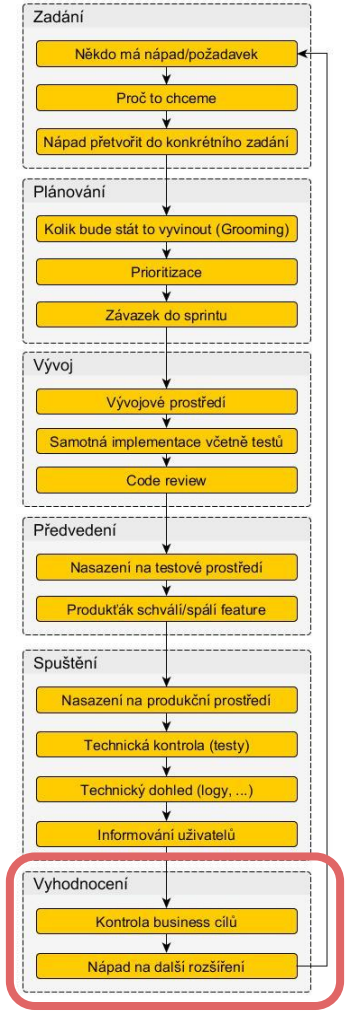
Marketing

- Je důležité dát světu vědět, že jsme přišli s nějakou novinkou
  - visuální průvodce novou funkcionalitou
  - changelog
  - newsletter



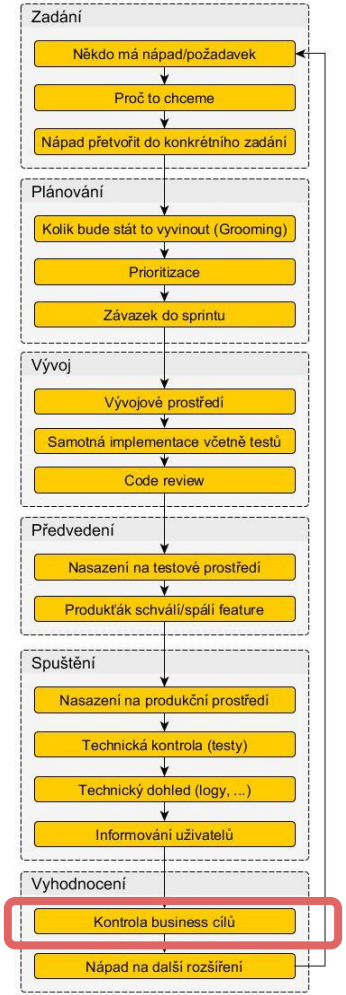
Vyhodnocení





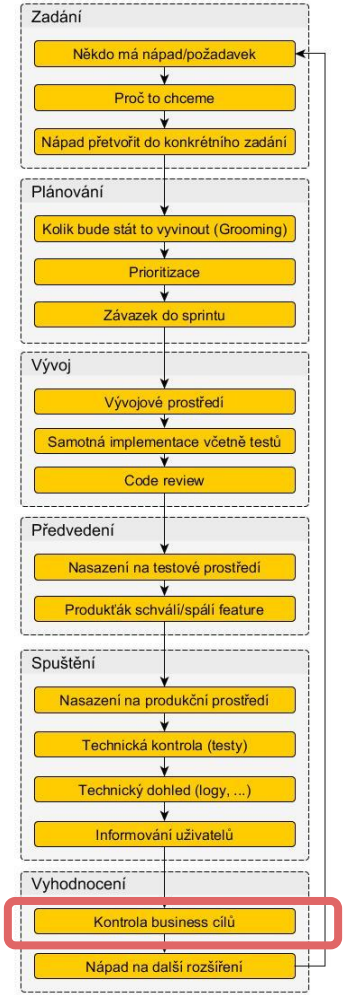
Kontrola business cílů

- Každá funkcionalita má mít nějaký měřitelný přínos.
- Po vyvinutí nové funkce je třeba zkontrolovat že tento přínos byl naplněn
- To že je aplikace po spuštění funkční a správně plní zadání ještě nemusí znamenat, že přináší očekávaný zisk



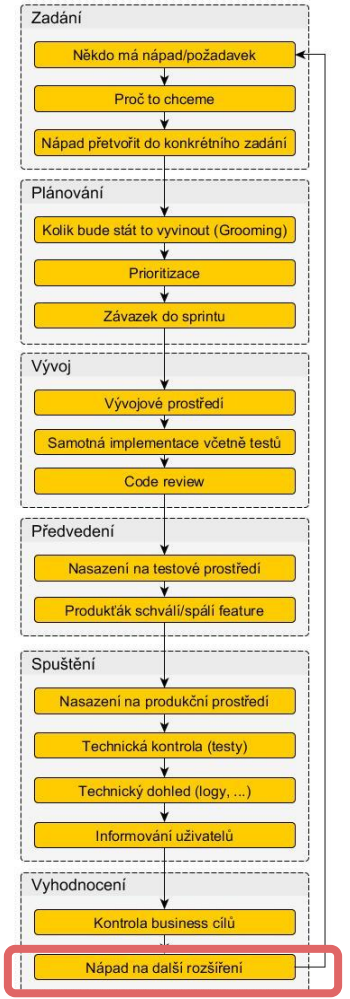
Kontrola business cílů

- Používají uživatelé novou funkcionalitu?
  - a/b testy
- Je nová funkcionalita výkonnější než původní?
  - měření Google Analytics, vlastní metriky
- Vývojem financování funkcionality nekončí
  - něco stojí provoz hardware
  - údržba sw - nová funkcionalita může být komplikovaná na pozdější úpravy
- Někdy může být výhodnější přiznat neúspěch



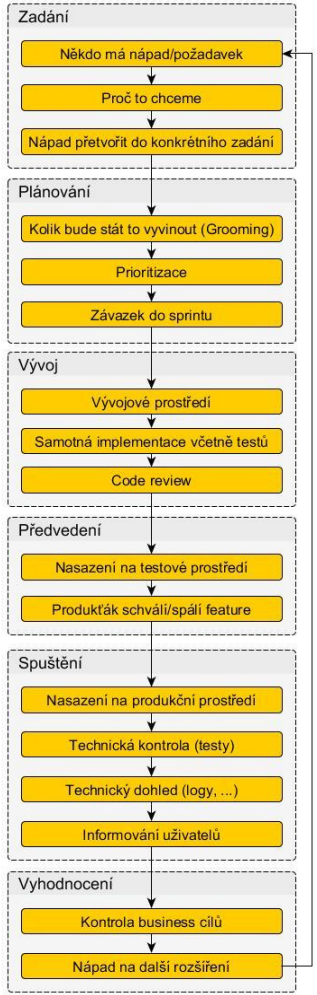
Jak rozvíjet aplikaci dál

- Sledování nasazené aplikace je jedním z faktorů, které ovlivňují skladbu další iterace vývoje.
- Funkcionalita se
  - osvědčila? - Můžeme dále rozvíjet a investovat
  - nikdo ji nepoužívá? - Můžeme zlepšit marketing
  - zákazníci ji používají, ale nevydělává? - Nejlépe upravit nastavení obchodních parametrů



Shrnutí

- Zadání - Plánování
  - Máme seznam formálních story
- Plánování - Vývoj
  - Vývojáři a produkťák jsou ve shodě v tom co se bude v následující sprintu dělat
- Vývoj - Předvedení
  - Na Testovém prostředí je funkční otestovaná aplikace
- Předvedení - Spuštění
  - Nová verze je připravena k instalaci na Produkci
- Spuštění - Vyhodnocení
  - Novinky na Produkci funguje bez chyb a máme podklady pro vyhodnocení



KONEC

Kontakty

Stažení prezentace: http://www.masicek.net

Jan Štěpánek: stepanek.j@gmail.com

Viktor Mašíček: viktor@masicek.net