SMP

Sběr, ověřování správnosti a správa požadavků. Dokumentace požadavků textově a modelováním pomocí UML. Firemní(business) procesy. (SMP)

Obsah

[Business analýza 3](#_Toc515893353)

[Business cíle: 4](#_Toc515893354)

[Business požadavky 4](#_Toc515893355)

[Funkční požadavky 4](#_Toc515893356)

[Kategorizace požadavků 5](#_Toc515893357)

[Kde dolovat požadavky 5](#_Toc515893358)

[Zainteresované osoby 5](#_Toc515893359)

[0věřován požadavků 6](#_Toc515893360)

[Obsah 6](#_Toc515893361)

[Forma 6](#_Toc515893362)

[Business procesy 6](#_Toc515893363)

[Techniky sběru požadavků 7](#_Toc515893364)

[Řídící metodiky 8](#_Toc515893365)

[Tradiční řízení 8](#_Toc515893366)

[Iterační řízení 8](#_Toc515893367)

[Agilní přístup 9](#_Toc515893368)

[Domů extrémní programování 9](#_Toc515893369)

[Unified proces 9](#_Toc515893370)

[Scrum 10](#_Toc515893371)

[Princip 10](#_Toc515893372)

[Procesy další části 11](#_Toc515893373)

[Diagramy: 12](#_Toc515893374)

[diagram aktivit/procesů 12](#_Toc515893375)

[Stavový diagram 12](#_Toc515893376)

[Diagram tříd 13](#_Toc515893377)

[diagram případů užití 14](#_Toc515893378)

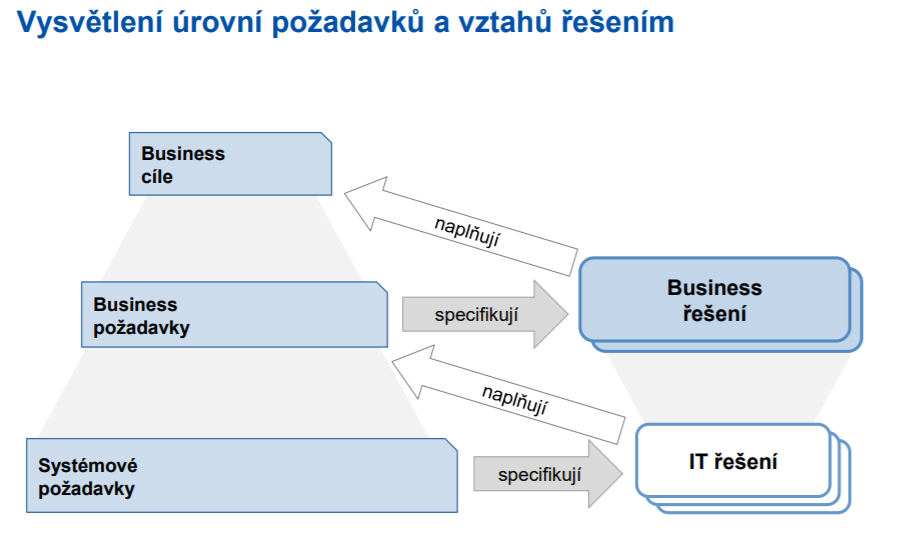
[<<extend>> a <<include>> 14](#_Toc515893379)

[diagram komponent 15](#_Toc515893380)

[diagram nasazení 16](#_Toc515893381)

[sekvenční diagram 17](#_Toc515893382)

# Business analýza



* BA je řada aktivit, které
  + Slouží k pochopení struktury, způsobu fungování a postupů v organizaci.
  + Definují business cíle a business požadavky.
  + Umožní doporučit řešení, které organizaci posune blíže k jejím cílům.
* Musíme si uvědomit
  + Problém / příležitost a jeho dopad do organizace.
  + Očekávané přínosy odstranění problému / využití příležitosti.
  + Doba nutná k řešení problému / využití příležitosti.
  + Náklady na řešení problému / využití příležitosti.
  + Náklady nic nedělání.
* Kdo je business analytik?
  + Sbírá potřeby a požadavky napříč organizací.
  + Identifikuje skutečné požadavky v organizaci (nejen touhy lidí/oddělení).
  + Zprostředkovává komunikaci mezi různými odděleními.
* Velmi vhodné je rozlišovat úrovně požadavků:
  + Business cíle
  + Business požadavky
  + Systémové požadavky
* Jak identifikujeme požadavek
  + **Shora-dolů:** plnění business cílů
  + **Zdola-nahoru**: problémy v procesu/funkcionalitě
  + **Od středního managementu**: potřeba dalších informací pro rozhodnutí nebo doplňkové funkcionality
  + **Od externích zdrojů**: potřeba zákazníka, regulátora apod.
* postup při business analýze

1. Business analytik **shromažďuje business potřeby** (a omezení) a business cíle.
2. Business potřeby a cíle se **transformují na business požadavky**.
3. Na základě požadavků (a omezení) vznikají **návrhy variant business řešení**.
4. Nakonec je vybráno **jedno řešení**, které je dále zpracováváno

* Co je tedy výsledkem business analýzy?
  + Pochopení struktury, způsobu fungování a postupů v organizaci.
  + Analýza variant řešení a návrh dalšího postupu.
  + Business požadavky.

## Business cíle:

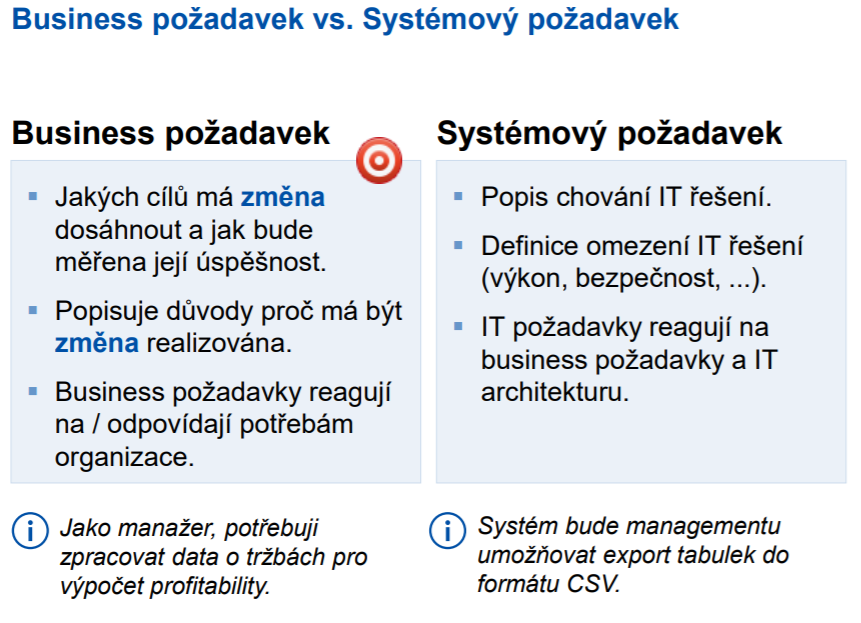
* Hlavní cíle projektu.
* Často definovány či schvalovány vysokým management firmy.
* Definuje na úrovni celého projektu:
  + CO se má změnit?
  + Proč se to má změnit? = přínosy.
  + Uspokojivý cílový stav = jak vyhodnotím výsledek/přínos

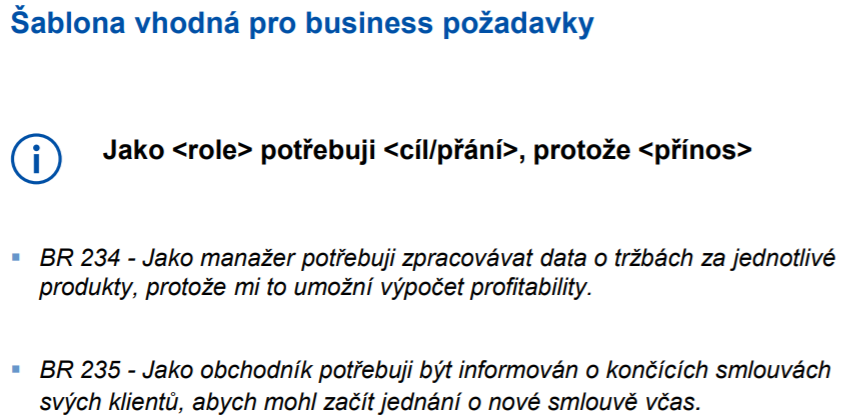
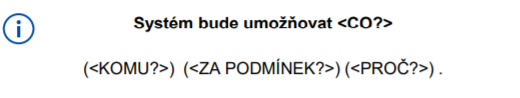
## Business požadavky

* Vznikají detailním rozpracováním jednoho nebo více souvisejících business cílů.
* Definují (detailněji než cíl):
  + CO se má změnit?
  + Proč se to má změnit? = přínosy.
  + Uspokojivý cílový stav = jak vyhodnotím výsledek/přínos.
* Můžou se vztahovat pouze k určité variantě řešení
* **Příklady:**
  + Je nutné snížit počet duplicitně vedených klientů, abychom mohli lépe cílit marketingové kampaně.
  + Práce se zrakovou zátěží musí být v zájmu omezení jejího nepříznivého vlivu na zdraví zaměstnance přerušována bezpečnostními přestávkami

## Funkční požadavky

* FR 3 Smazání směny: Systém umožní manažerovi smazat směnu, která ovšem ještě neproběhla.
* FR 4 Úprava směny: Systém umožní manažerovi úpravu již vytvořené směny.
* FR 5 Okomentování směny: Systém umožní manažerovi přidání komentáře ke směně



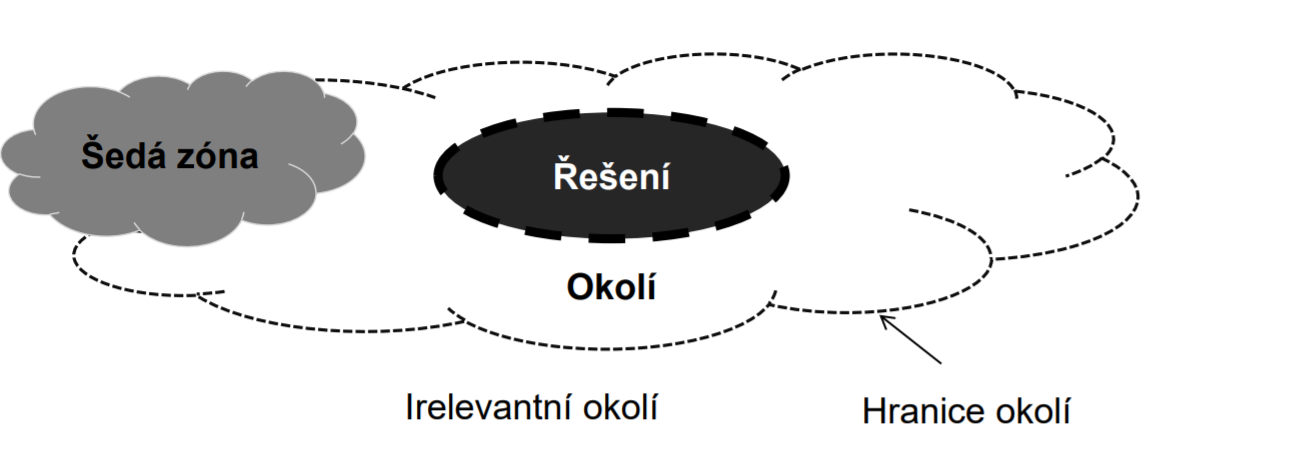


## Kategorizace požadavků

* A. **Functional Requiremen**t (FR): Funkční požadavky
  + Definují co systém bude uživatelům umožňovat.
  + (schopnosti systému)
* B. **Non-functional Requirement** (NFR): Kvalitativní požadavky a omezení
  + uživatelskou přívětivost, spolehlivost, výkon, bezpečnost a zabezpečení, rozšiřitelnost, licencování (a další právní aspekty), požadované technologie a rozhraní
  + vlastnosti systému
* C. **Transitional Requirement** : Požadavky na zavedení řešení
  + tvorbu uživatelské/administrátorské dokumentace, školení klíčových a ostatních uživatelů, požadavky na migraci dat
* Nutné vlastnosti (Dissatisfiers – „Rozlaďovače“)
  + Vlastnosti/požadavky, které nebyly stakeholderem ani zmiňovány, protože je považoval za zcela zřejmé
* Požadované vlastnosti (Satisfiers)
  + Explicitně zmíněné vlastnosti/požadavky.
* Zbytné vlastnosti (Delighters – „Nadchávače“)
  + Vlastnosti systému, které stakeholder neočekával, ale po jejich objevení/používání je považuje za příjemné vylepšení.

## Kde dolovat požadavky

* Zainteresované osoby a instituce (objednatel, uživatel, ...)
* Dokumenty (řídící dokumentace, regulace a legislativa, návody a manuály, ...)
* Procesy
* Stávající systémy
* Konkurence a trh



## Zainteresované osoby

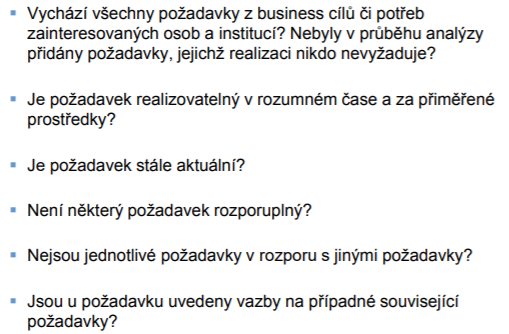
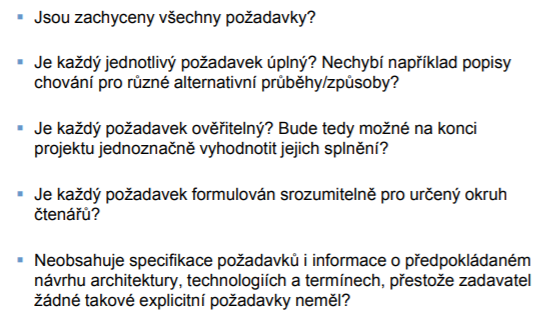
Zainteresovaná osoba (nebo instituce), která má přímý nebo nepřímý vliv na definici požadavků na systém:

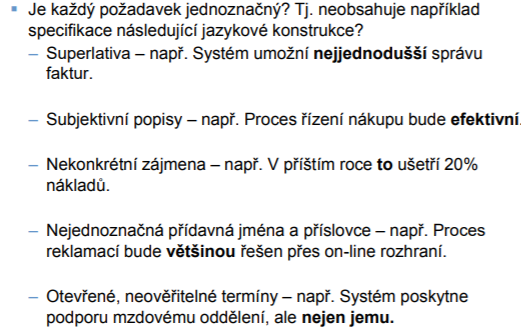
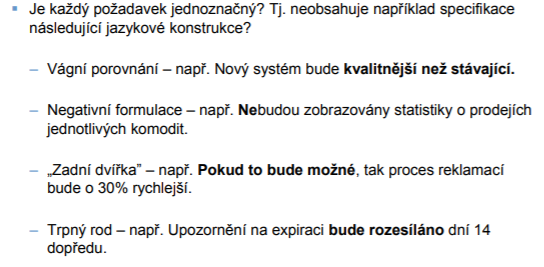
* Zákazníci – Customers
* Odborníci na návrh – IT experts
* Odborníci na testování – Testers

Odborníci na bezpečnost – Security experts

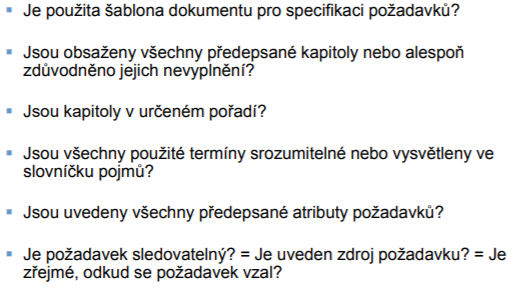
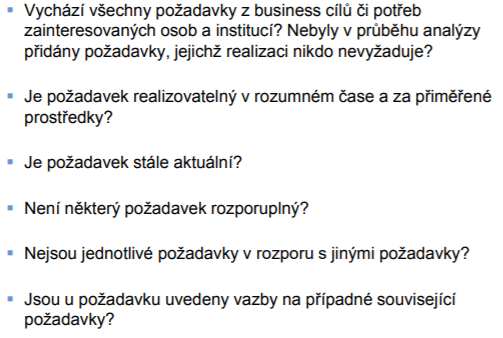
* Projektový management – Project managers
* Dodavatelé – Suppliers
* Znalci konkurence – Competetive intelligence experts
* Regulátor a jiné organizace - Regulator/ Institutions

## Ověřován požadavků

Obsah:



### Forma



## Business procesy

* Obchodní/firemní postupy.
* Po částech uspořádaná množina aktivit, které společně realizují podnikatelský nebo strategický cíl.
* Proč modelovat procesy
  + Pro potřeby tvorby nabídky.
  + Model (business) procesů slouží jako podklad pro odsouhlasení rozsahu aplikace klientem. Tedy identifikace procesů podporovaných a nepodporovaných (v praxi rozlišuji barvami) budoucím systémem.
  + Pro potřeby detailní analýzy systému
  + Pro vylepšení (reengineering) procesů.
  + Pro potřeby nasazení systému.
    - pro testery – tester může testovat i business logiku
* Typy procesu – můžou být **AS-IS / TO-BE**
  + **Hlavní procesy** vytvářejí hodnotu nebo užitek pro zákazníka organizace, vytvářejí výrobek nebo službu
  + **Podpůrné procesy** jsou všechny procesy, jejichž jediným cílem je zajistit fungování hlavních procesů a organizace
  + **Řídicí procesy** a činnosti jsou všechny aktivity, které koordinují, řídí, organizují a plánují vše ostatní

# Techniky sběru požadavků

Vlastní nápady, Analýza rozhraní, Prototypování, Dotazníkový průzkum, Odborná konzultace

## Skupinová jednání

**výhody**:

* + Při dobrém moderátorovi druhá nejúčinnější metoda
  + Adaptibilní, dovede odhalit nečekané skutečnosti
  + Synergické efekty větší než při běžném interview díky osobním interakcím
  + Lze využít pro vybudování týmového klimatu a pocitu odpovědnosti za projekt
  + Menší nebezpečí opomenutí
  + ± Větší rychlost

**nevýhody**:

* + Křiklouni ovládnou jednání
  + Obtížná kontrola kvality (lze zlepšit pomocí IS projektu)
  + Dosti pracné
  + Menší produktivita
  + Organizační problémy

## Individuální rozhovory

**výhody**:

* + Při dobrém moderátorovi vůbec nejúčinnější metoda sběru požadavků
  + Adaptibilní, dovede odhalit nečekané skutečnosti

**nevýhody**:

* + Časově náročné – Lze kontaktovat jen omezený počet respondentů
  + Obtížná kontrola kvality (lze zlepšit pomocí IS projektu)
  + Snadno se něco opomene
  + Poměrně pracné

## Analýza existujících dokumentů

**výhody**:

* + sleduje se to co existuje
  + dají se zjistit důležité typy činností a dat
  + levné
  + dá se použít jako inspirace

**nevýhody**:

* + často jde o zastaralé, a i nepotřebné dokumenty
  + muže inspirovat zastaralá řešení a nelogičnosti
  + pro dodavatele to snižuje možnost uplatnění existujících aplikací

## Brainstorming

**výhody**:

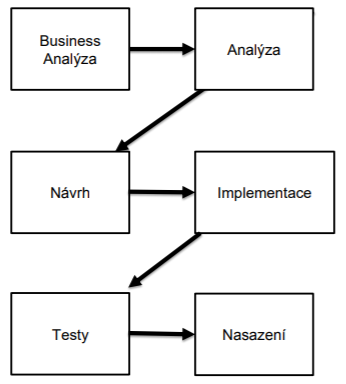
* + při dobrém moderátorovi a dobré přípravě lidí velmi účinná metoda hledání nových požadavků a nápadů
  + adaptibilní
  + synergické efekty

**nevýhody**:

* + nebezpečí plkání
  + obtížná kontrola kvality (lze zlepšit pomocí IS podporujícího vývoj projektu)
  + snadno se něco opomene
  + jen pro etapu vizí a začátek specifikace požadavků

# Řídící metodiky

## Tradiční řízení – Waterfall



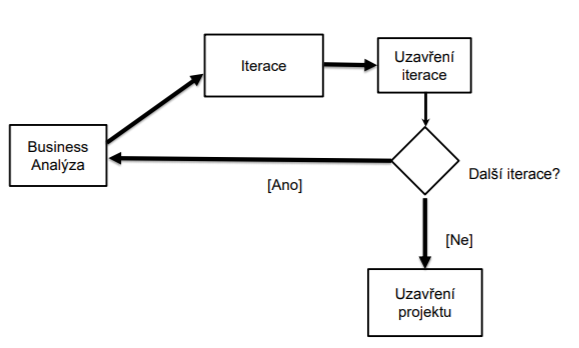
**Silné stránky**

* Už od začátku je jasný plán => víme co bude
* snadné řízení zdrojů (paralelismus projektů)
* Zvládne to i začátečník/průměrný zaměstnanec (rozdíl oproti agilním)
* Není třeba aby tým byl na jednom místě – analýza u nás, implementace v Malajsii
* Snadné předání externímu dodavateli

**Slabé stránky**

* Špatně reaguje na změnu
* Stojí hodně peněz – veškeré chyby se projeví až při předání (akceptačním testu) a pak je drahé chyby opravit
* Dlouho trvá, než zákazník vidí nějaký výstup
* Vyžaduje detailní plán a specifikaci – každé přeplánování je drahé
* Sleduje procesy, které není možné změnit
* Není zaměřen na přínos klientovi – je zaměřen na dodání v nějaký čas – dodání podle specifikace není často totožné s tím co zákazníkovi pomůže – potřeby vs. přání

## Iterační řízení



**Charakter**

* Častý sběr požadavků na změnu
* Tým v jedné lokalitě
* Iterace 2-4 týdny
* Část řešení není známa
  + nejsou detaily -> změny
* Zpětná vazba s klientem
* Jednoduché/lehké zapojení klienta

**Nevýhody**

* Zapojení klienta
* Týmy v jedné lokalitě
* Není vždy jasné, jak to dopadne

**Potřeby**

* Malý tým
* Zkušení pracovníci

## Agilní přístup

* **Cíl**: Spokojený zákazník, spokojený programátor
* **Kdy**: Zkušení programátoři, Očekává se hodně změn, Malý tým
* **Motto/manifest**:
  + Člověk a komunikace je nad procesy a nástroje
  + Fungující software je nad vyčerpávající dokumentaci
  + Spolupráce se zákazníkem je nad přesně sjednanou smlouvou
  + Reakce na změnu je nad přesné dodržování plánu (Ale zas má agile pravidlo – nezasahovat do iterace. Proto krátké iterace)
* **Kdy ne**: Velké programátorské týmy, Dislokované týmy, Nutnost fixace projektového trojúhelníku
  + Velký tým ve velké kanceláři… nemůže dělat meetingy... Horší komunikace
  + Dislokované týmy – nejsou meetingy
  + Agile – se nedá přikázat, musí se aplikovat všechno - =
  + Kritické systémy – vyžadují dokumentaci, většinou jsou předem jasné požadavky

## Domů extrémní programování

* Jedná se o tradiční činnosti, které jsou však dovedeny do extrému.
* Zákazník je vždy k dispozici
* Coding standards
* Zaměření na okamžitou přidanou hodnotu
* Vydávání malých verzí

## Unified proces

* vhodnější spíš pro rozsáhlejší projekty a větší vývojové týmy,
* iterativní vývoj softwaru
* počítá s častou změnou požadavků
* průběžné zajišťování a ověřování kvality
* Další iterace/fáze projektu začne jen v případě splnění kritérií.
* řízení změn

**zahájení**: vývojáři definují účel a rozsah projektu a jeho obchodní kontext. Výstup: Vize, Plán, Základní požadavky

**projektování**: analýza potřeb projektu a zákazníka a definování základu architektury. zpřesňování představy/požadavků. Sestavení projekčního týmu Výstup: Prototyp, Use case (80%), Návrh(10%), Datový model, Plán testů, Plán projektu,

**realizace**: na vývoj designu aplikace a tvorba zdrojových kódů. dokončení návrhu, první verze SW, návrh testů

**předání**: dochází k předání projektu – buď zákazníkovi nebo do dalšího vývojového cyklu

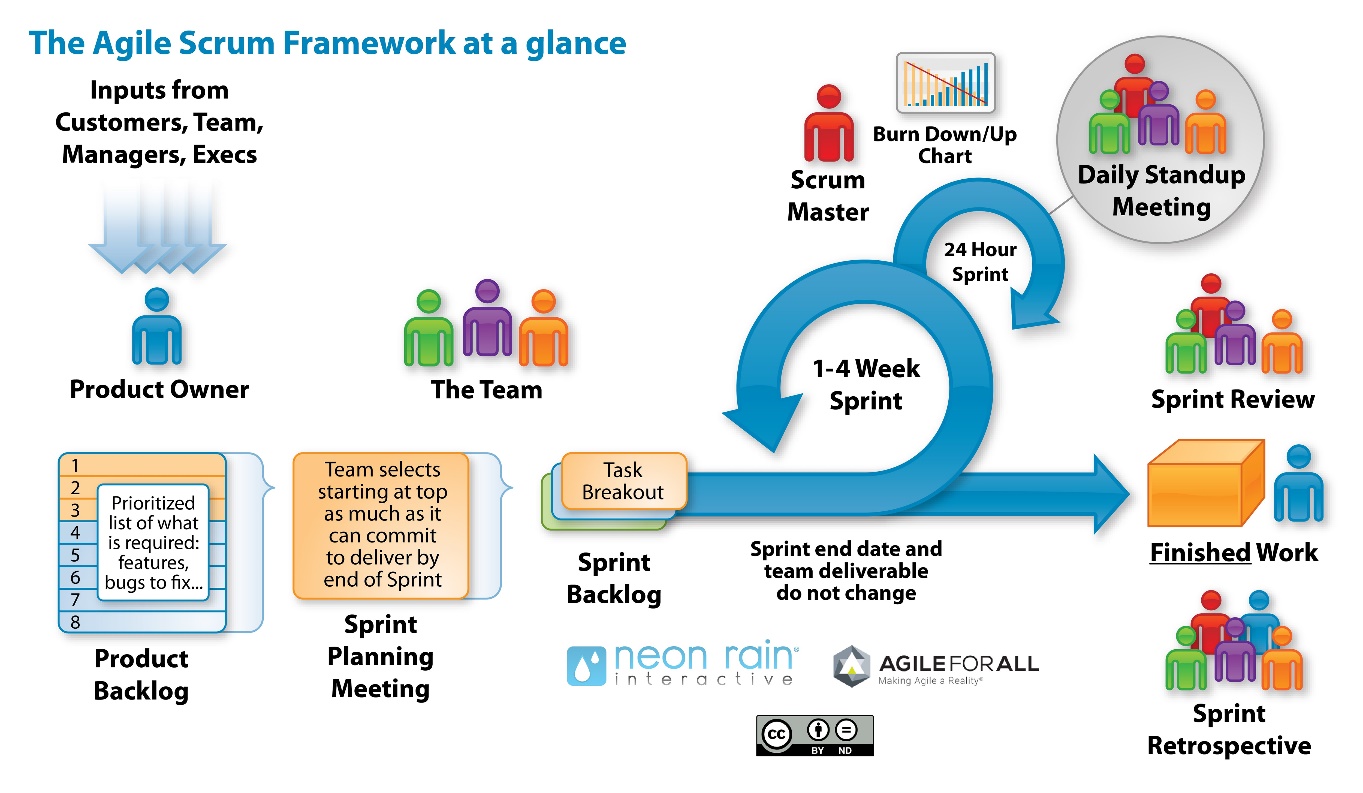


Uvnitř každé fáze dále probíhají iterace; jejich počet závisí na konkrétních potřebách týmu a rozsahu projektu. Každá iterace je podrobně rozpracována; je stanoveno, kdo ji má dělat (roles), jak ji má dělat (activities) a co má být jejím výsledkem (artifacts).fied proces

## Scrum

* iterativní a inkrementální metodologie agilního vývoje softwaru používaná na řízení produktového vývoje.
* zpochybňuje předpoklady "tradičního, sekvenčního přístupu" k vývoji produktu
* Klíčový princip scrumu je jeho pochopení, že během projektu mohou zákazníci změnit názor o tom, co chtějí a potřebují
* Scrum používá empirický přístup, podle kterého problém nelze zcela pochopit nebo definovat a proto se soustředí na maximální schopnost týmu rychle dodat a reagovat na nové požadavky

### Princip



#### Role

**Vlastník produkt:** Vlastník produktu reprezentuje zainteresované subjekty a je hlasem zákazníka.

* demonstrovat řešeni pro klíčové stakeholdery, kteří nebyli na iteračním demu
* ohlašovat uvedení nových verzí
* komunikovat stav týmu
* organizovat milníkové přehledy
* vzdělávat v procesu vývoje
* dohadovat priority, rozsah, financování a rozvrh
* ujistit se, že produktové testy jsou viditelné, transparentní a jasné

**Vývojový team**: Je odpovědný za dodání potenciálně použitelných inkrementů (potentially shippable increments - PSIs) produktu na konci každého sprintu (cíl sprintu). Team je složen z 3-9 jednotlivců, kteří dělají aktuální práci (analýza, design, vývoj, test, technická komunikace, dokument atd.).

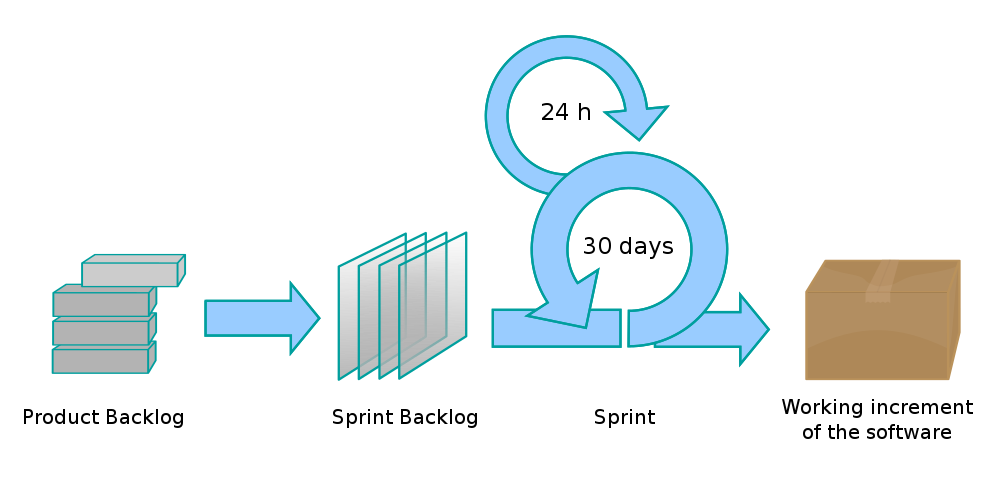
**Scrum master:** je odpovědný za odstranění překážek teamu na dodání produktových cílů. Scrum master není tradiční team leader nebo projektový manažer, ale koná jako prostředník mezi teamem a jakýmikoli negativními vlivy. Často organizuje schůzky a povzbuzuje tým k zlepšení.

### Procesy další části

**Product Backlog:** Seznam všech požadavků klienta seřazený podle priority. Jedná se jednoduše popsané požadavky, nikoliv dokumentace

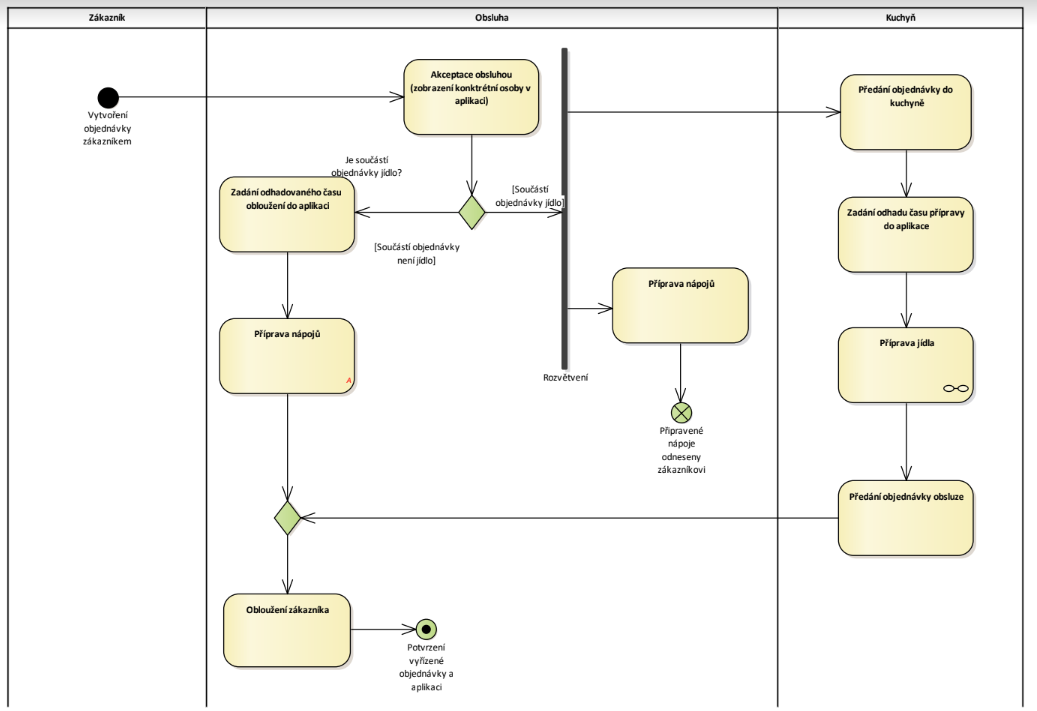
**Sprint**: (iterace) je základní jednotka vývoje ve scrumu. Každý sprint začíná eventem plánování sprintu, cílem, kterého je definovat úkoly sprintu, kde je definována práce sprintu a odhadnut závazek pro cíl sprintu. Každý sprint končí recenzí sprintu (sprint review) a retrospektivou, kde je reportován progres pro stakeholdery a definují se úlohy na zlepšení pro další sprinty.

**Sprint backlog:** část úkolů zvolené z Product Backlogu pro stávající Sprint. Dílčí cíle jednoho inkrementu



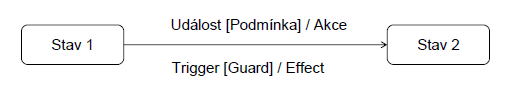
# Diagramy:

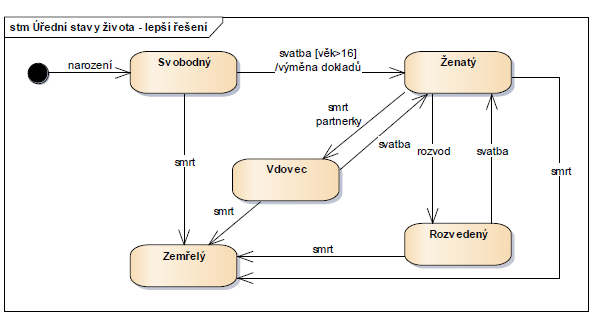
## Diagram aktivit/procesů

* Slouží k zachycení business procesů **(AS-IS** a variant **TO-BE)** v rámci hledání business řešení.
* Je rychleji pochopitelný než textový zápis
* **Prvky diagramu:** začátek, konec, pool/swimlane, aktivita, rozhodnutí

## Stavový diagram

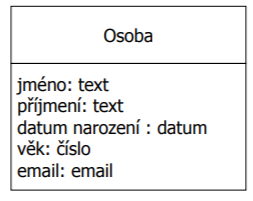
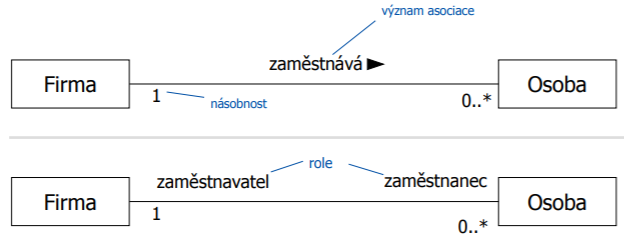
* Slouží k modelování **životního** **cyklu** entit
* **Prvky diagramu**: začátek, konec (ne vždy), stavy, přechody
* na základě nějaké podmínky se při události změní stav entity ze stavu 1 do stavu 2

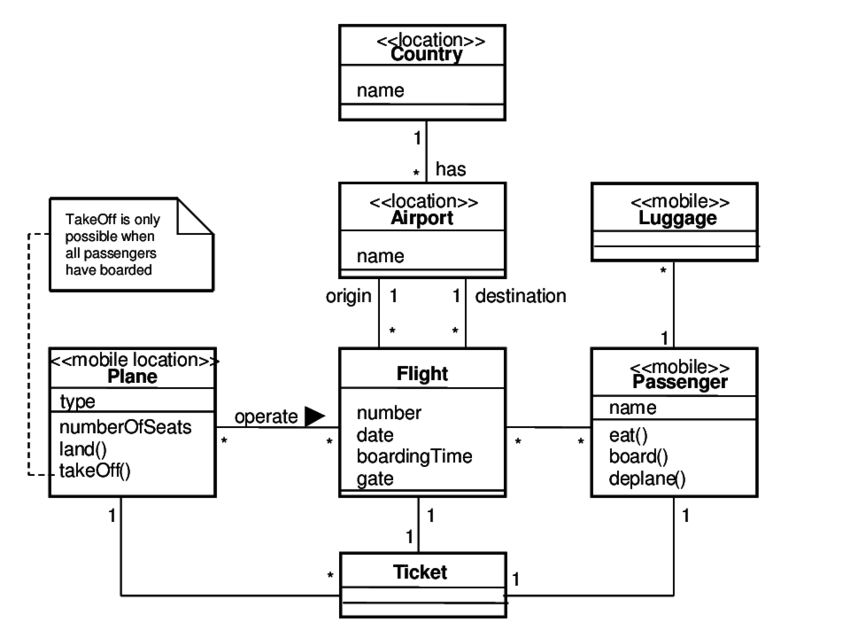




## Diagram tříd

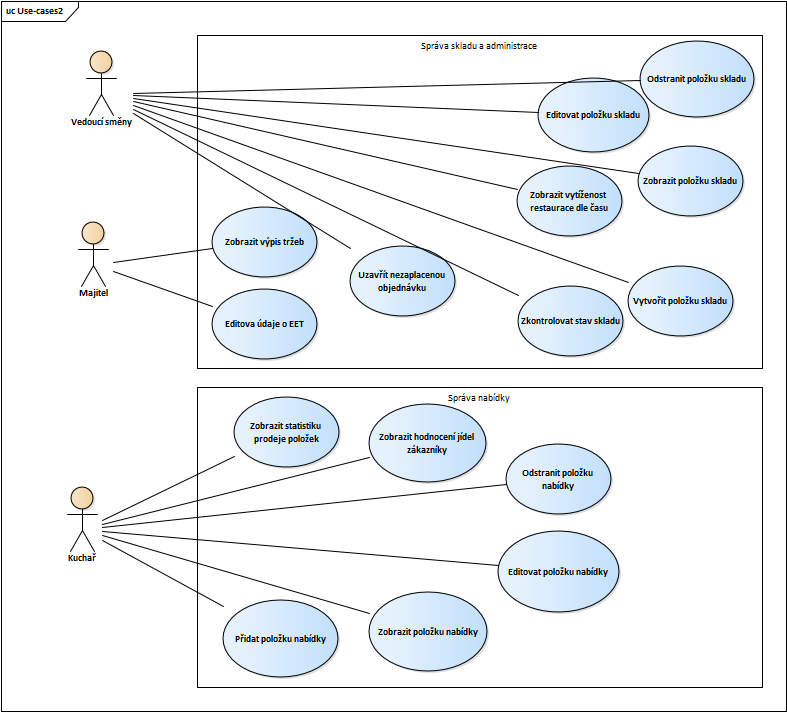
* Slouží na definici business entit.
* **Prvky diagramu**: Entity s atributy, vztahy s kardinalitami





## Diagram případů užití – Use case diagram

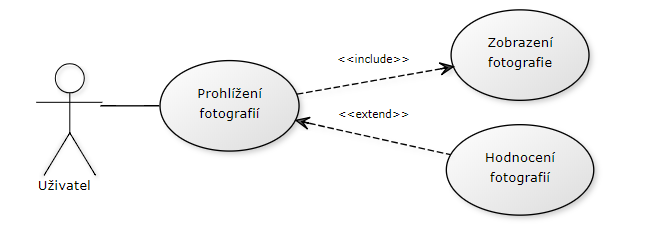
* Vyjadřuje, kdo bude jakým způsobem používat systém = uživatele systému a jejich práva
* **Prvky diagramu**: Aktéři (kdo bude systém používat), případy užití (co bude systém umožňovat)



### <<extend>> a <<include>>

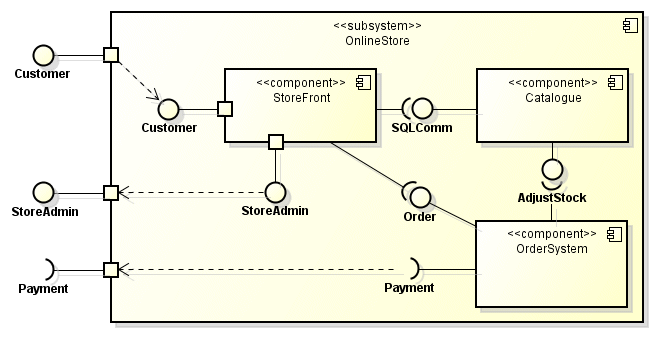
* Tyto vazby “odkazují” na jiné use casy

|  |  |  |
| --- | --- | --- |
|  | Vazba <<include>> | Vazba <<extend>> |
| Zamezuje duplicitám ve scénářích | ✓ | ✓ |
| Vyvolání | vždy když je vyvolán nadřazený use case | může, nemusí být vyvolán |
| Kontextová závislost | funguje jako use case i bez nadřazeného | bez nadřazeného kontextu use casu nemá význam |



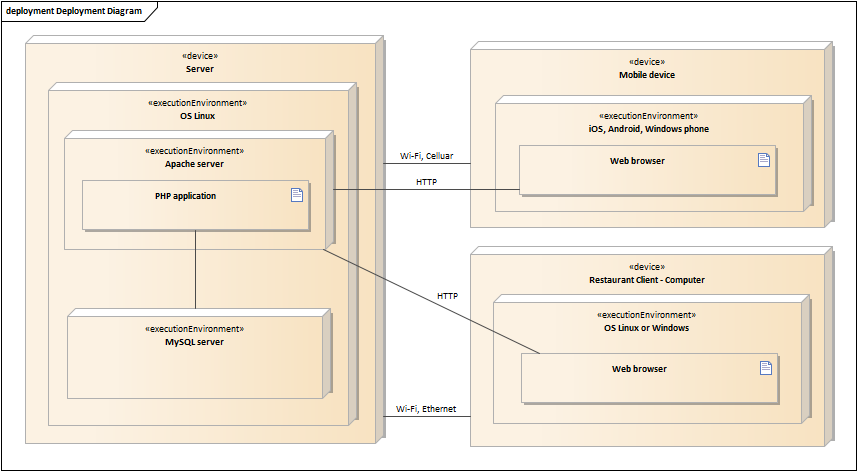
## Diagram komponent

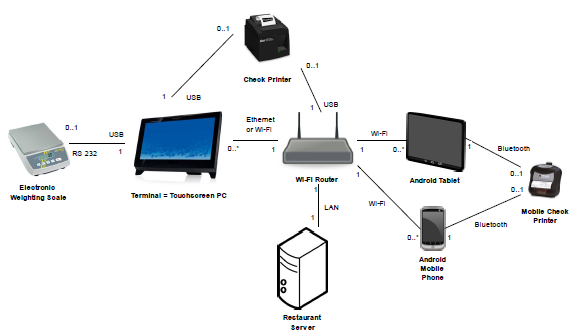
* popisuje logickou architekturu systému (rozdělení systému na subsystémy/moduly)
* **prvky diagramu**: komponenta, poskytované rozhraní, požadované rozhraní



## Diagram nasazení

* popisuje HW architekturu systému a nasazení SW na nej.
* **prvky diagramu**: uzel, asociace





## Sekvenční diagram

* ukazují kdo (instance tříd) se podílí na realizaci akce, jaké správy a parametry si objekty vyměňují a v jakém pořadí.

