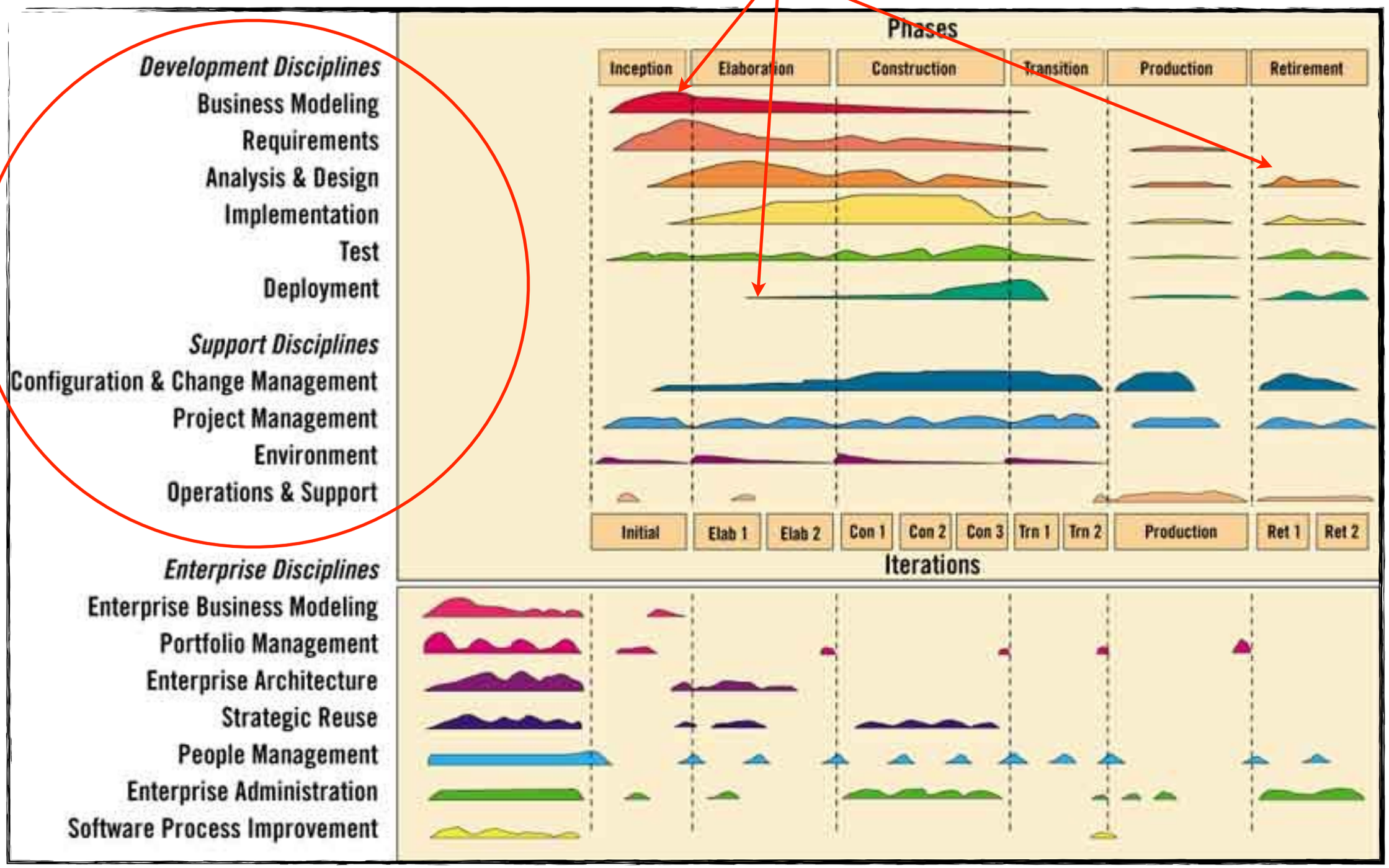
RUP - Disciplíny

Jaroslav Žáček
jaroslav.zacek@osu.cz
http://www1.osu.cz/~zacek/infs1/

Disciplíny

Množství disciplíny v dané iteraci



Disciplíny podle RUP

Šest základních:

- **Business modeling** - pro pochopení problémové domény
- **Requirements** - zjištění požadavků ve formě scénářů (Use Case)
- **Analysis & design** - co se bude tvořit a jak se to bude tvořit
- **Implementation** - kódování a psaní testů
- **Test** - ověření a hodnocení kvality produktu
- **Deployment** - doručení SW všem uživatelům

Disciplíny podle RUP

...a tři podpůrné:

- **Configuration and change** management - správa změn a konfigurací
- **Project management** - plánování, řízení a monitorování
- **Environment** - konfigurace procesu, nástrojů na podporu týmu

Co je to disciplína

- Disciplína není jen fáze vodopádu!
- Disciplína je seskupení aktivit vykonávaných v různých fázích projektu.
- Každou (v praxi většinu) z těchto disciplín provádíme v každé iteraci každé fáze iterativně založeného projektu.
- Disciplíny jsou souhrnem, kolekcí úkolů a aktivit týkajících se konkrétní oblasti zájmu (požadavky, analýza, testování, …)
- Objem prací z každé disciplíny v daných fázích RUP definuje množství plochy pod křivkou v modelu (viz RUP model RUPu)

Business modeling

PARTNERS

I amsterdam.

amazon

Heineken

boll.com®

AOL

ProSieben

KEY ACTIVITIES

STOCK MANAGEMENT

SEARCH

ARTIST MANAGEMENT

CONTRACT MANAGEMENT

FUND MANAGEMENT

COMMUNITY MANAGEMENT

PRODUCTION & PUBLISHING

KEY RESOURCES

SOFTWARE & RIGHTS

DISTRIBUTION

KNOWHOW MUSIC-INDUSTRY

VALUE PROPOSITION

"FAN-FUNDED-MUSIC"

CLIENT RELATION

WWW.SELLABAND.COM

WWW.SELLABAND.DE

WWW.SELLABAND.JP

WWW.SELLABAND.COM

W³.SELLABAND.COM
W³.AMAZON.COM
W³.HEINEKEN.COM
-ENTERTAINMENT.NL

CLUBS

STORE

NEWS LETTERS

DISTRIBUTION & COMMUNICATION

CLIENTS

BELIEVERS !

105 COUNTRIES

INDEPENDENT

MUSIC-BUYER

REVENUE

AIRPLAY

MUSIC SALES

SPONSORING

$10,-

INTREST

TRANSACTION & HANDLING

COSTS

€ SELLABAND

€ PRODUCTION

€ DISTRIBUTION

€ HANDLING & MARKETING

€ SOFTWARE

€ PLATFORM

SELLABAND

OFFERED TO SELLABAND BY JAM, BUSINESS MODELS INC. AND PWC

Gillette: Razors & Blades

KEY PARTNERS

- Manufacturers
- Retailers

KEY ACTIVITIES

- Marketing
- R&D
- Logistics

KEY RESOURCES

- Brand
- Patents

VALUE PROPOSITION

- Razor handle
- Blades

RELATIONSHIPS

- Built-in "Lock-in"

CHANNELS

- Retail

CUSTOMER SEGMENT

- Mass customers

COST STRUCTURE

- Marketing
- Logistics
- Manufacturing
- R&D

REVENUE STREAMS

- 1x handle purchase
- Frequent blade replacements

Flickr: Photo Sharing

KEY PARTNERS

Yahoo!

KEY ACTIVITIES

Platform management

Selling

KEY RESOURCES

Flickr platform brand

VALUE PROPOSITION

Free basic photo sharing

Premium photo sharing

RELATIONSHIPS

Mass customized

CHANNELS

Flickr.com

Yahoo.com

CUSTOMER SEGMENT

Casual users

High volume users

COST STRUCTURE

Platform develpment

Storage cost

REVENUE STREAMS

Free limited basic account

Annual subscription pro account

Skype

KEY PARTNERS

- Payment Providers
- Distribution Partners
- Telco Partners

KEY ACTIVITIES

- Software Develoment

KEY RESOURCES

- Software Developers
- Software

VALUE PROPOSITION

- Free Internet & Video Calling
- Cheap Calls to Phones (SKYPEOUT)

RELATIONSHIPS

- Mass Customers

CHANNELS

- Skype.com
- Headset Partnerships

CUSTOMER SEGMENT

- Web Users Globally
- People Who Want to Call Phones

COST STRUCTURE

- Software Development
- Complaint Management

REVENUE STREAMS

- SKYPEOUT Prepaid or Subscription
- Hardware Sales

The Business Model Canvas

Designed for:

Designed by:

On: *dd/mm/yyyy*

Iteration #

Problem
top 3 problems

Solution
top 3 features

Unique value proposition
single, clean, compelling message that states why you are different and worth buying

Unfair advantage
can't be easily copied or bought

Customer Segments
target customers

Key metrics
key activities you measure

Channels
path to customers

Cost Structure
What are the most important costs inherent in our business model?
Which Key Resources are most expensive?
Which Key Activities are most expensive?

Revenue Streams
For what value are our customers really willing to pay?
For what do they currently pay?
How are they currently paying?
How would they prefer to pay?
How much does each Revenue Stream contribute to overall revenues?

Podnikový proces

- Je chápán jako souhrn činností přeměňující za pomoci lidí a dalších nástrojů vstupy na určité výstupy, které mají hodnotu pro zákazníky nebo jiné procesy.

- Podnikový proces se provádí za účelem splnění určitého podnikového cíle (business goal).

Modelování podnikových procesů

- není třeba vykonávat v rámci každého projektu (např. v případě nové verze produktu nebo rozšíření stávající verze)

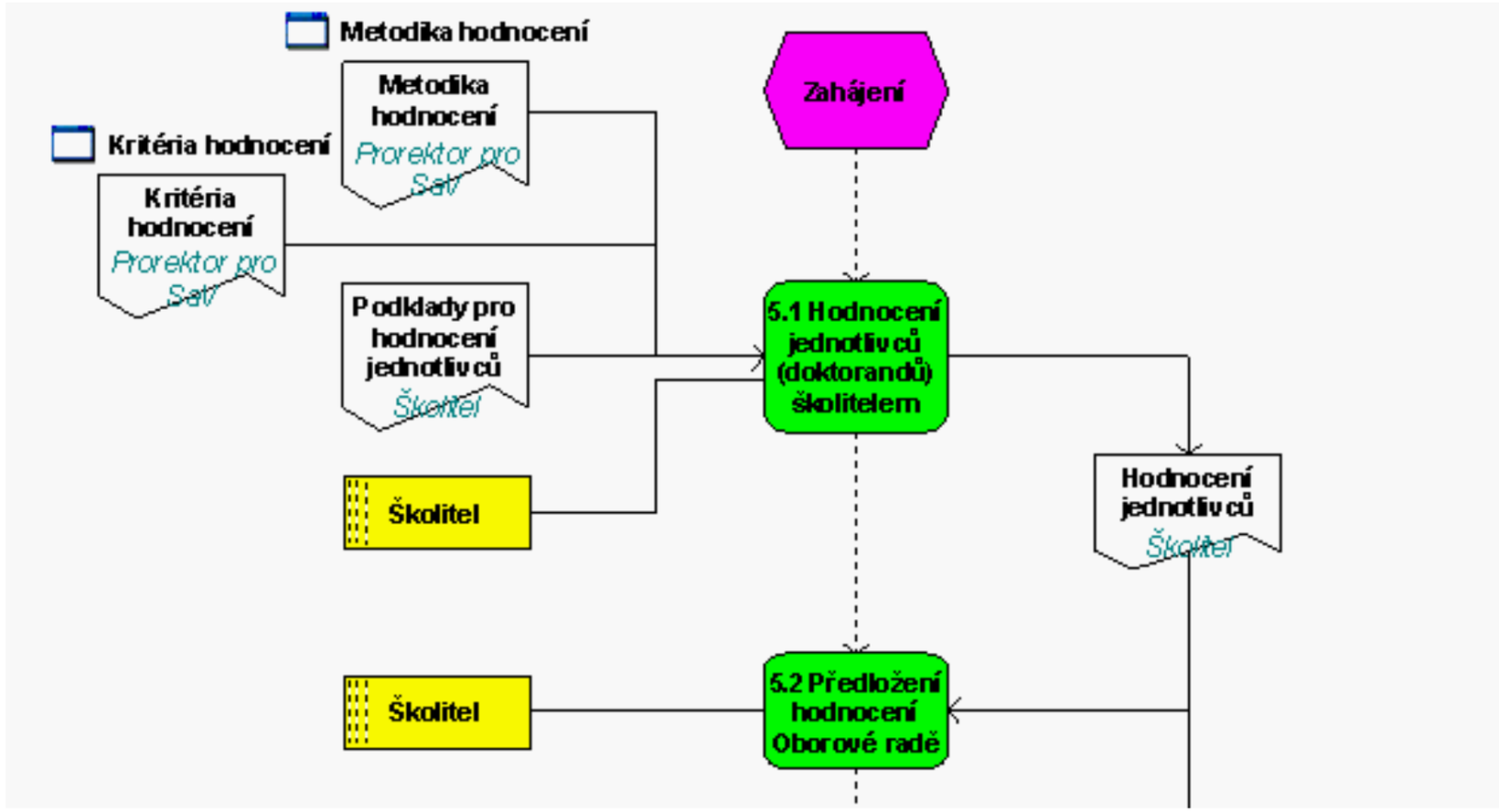
- procesní či doménový model (+ slovník pojmů) je velmi důležitý také pro potřebu údržby – porozumění, proč a k čemu uživatel aplikaci potřebuje

- Techniky pro pochopení problémové domény:

  ✓ Stínování zaměstnanců (sledování jejich práce, dotazy)

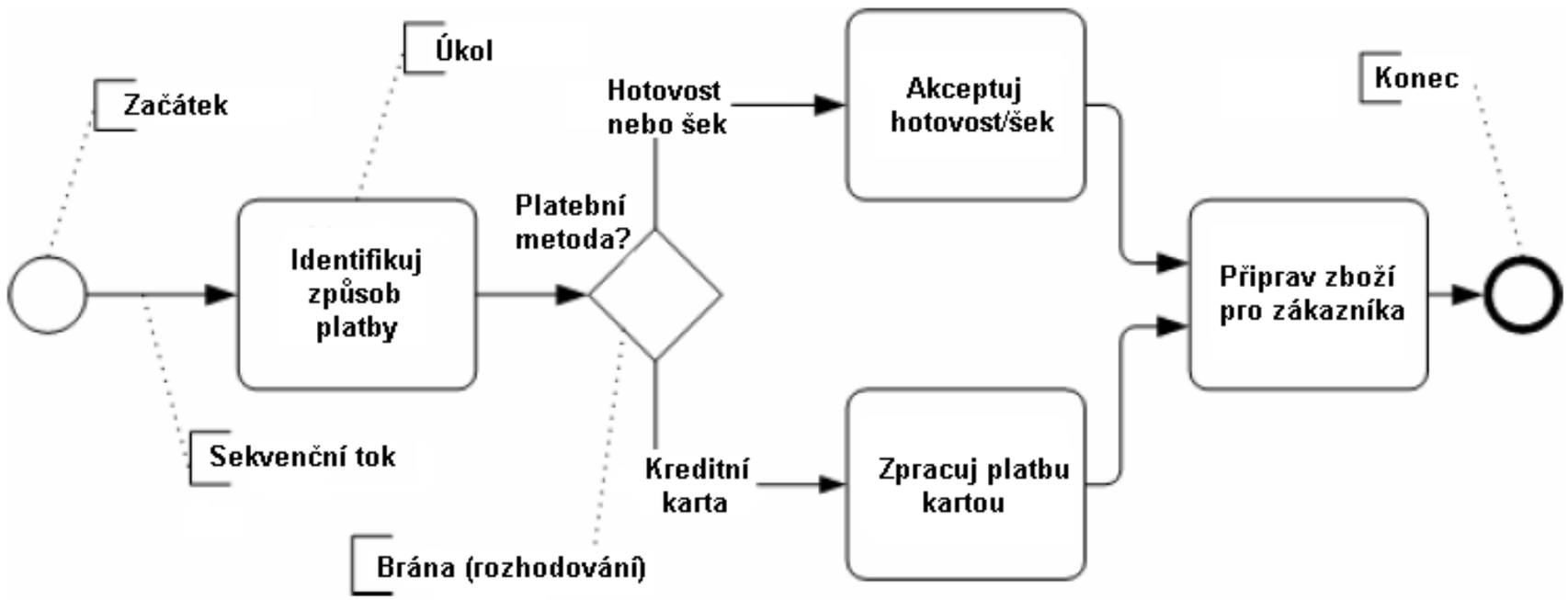
  ✓ Použití byznys konzultantů

  ✓ Tvorba jednoduchých prototypů a jejich demonstrace a diskuse nad nimi s budoucími uživateli či s byznys konzultanty.



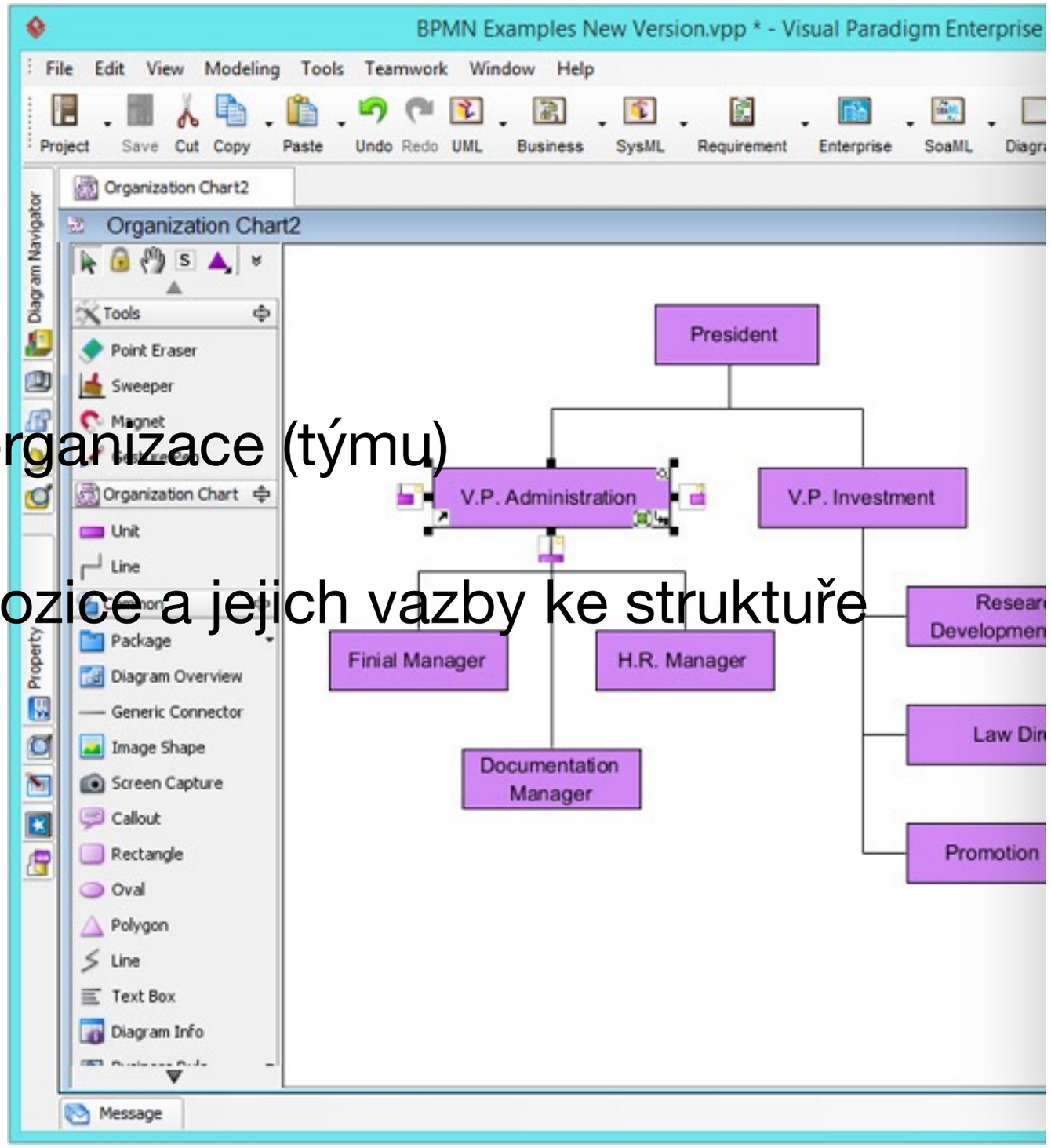
Notace EPC

Notace BPMN



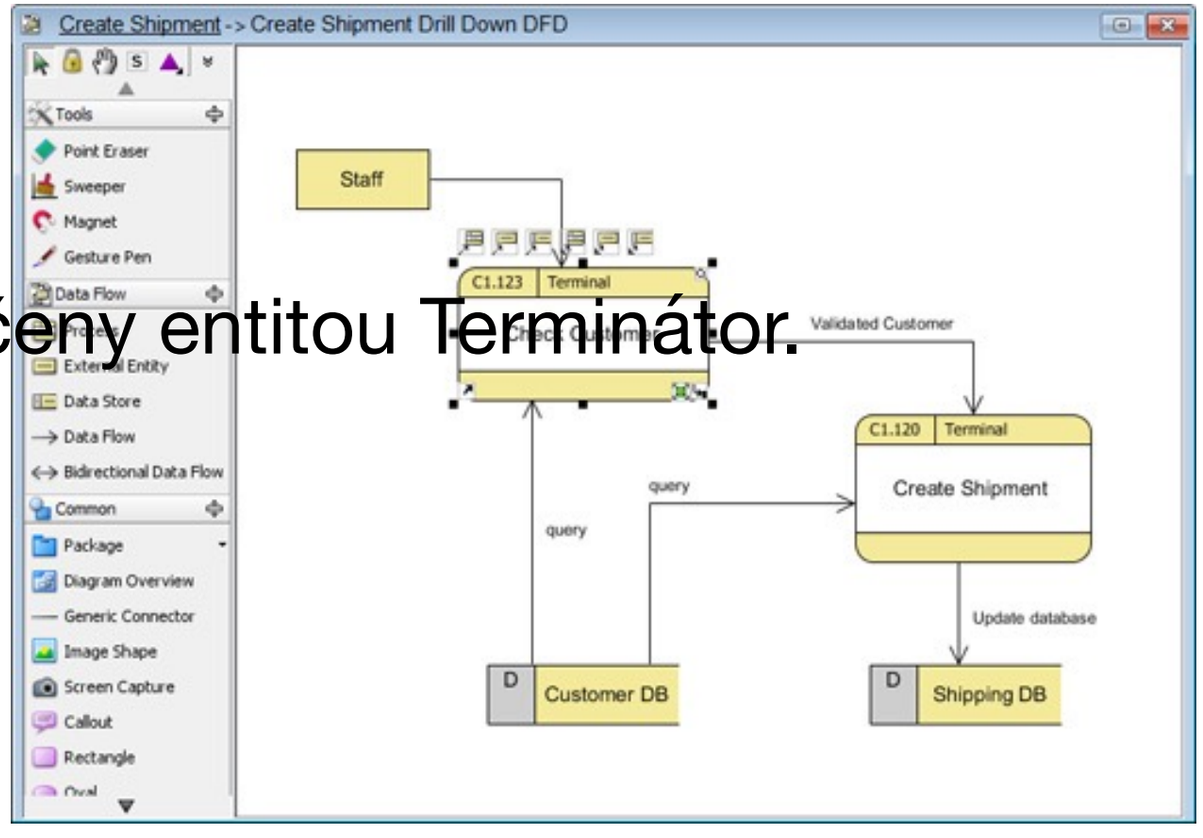
Organization chart

- Pro definování struktury organizace (týmu)
- Zachycuje lidi, pracovní pozice a jejich vazby ke struktuře podniku



Data flow diagram

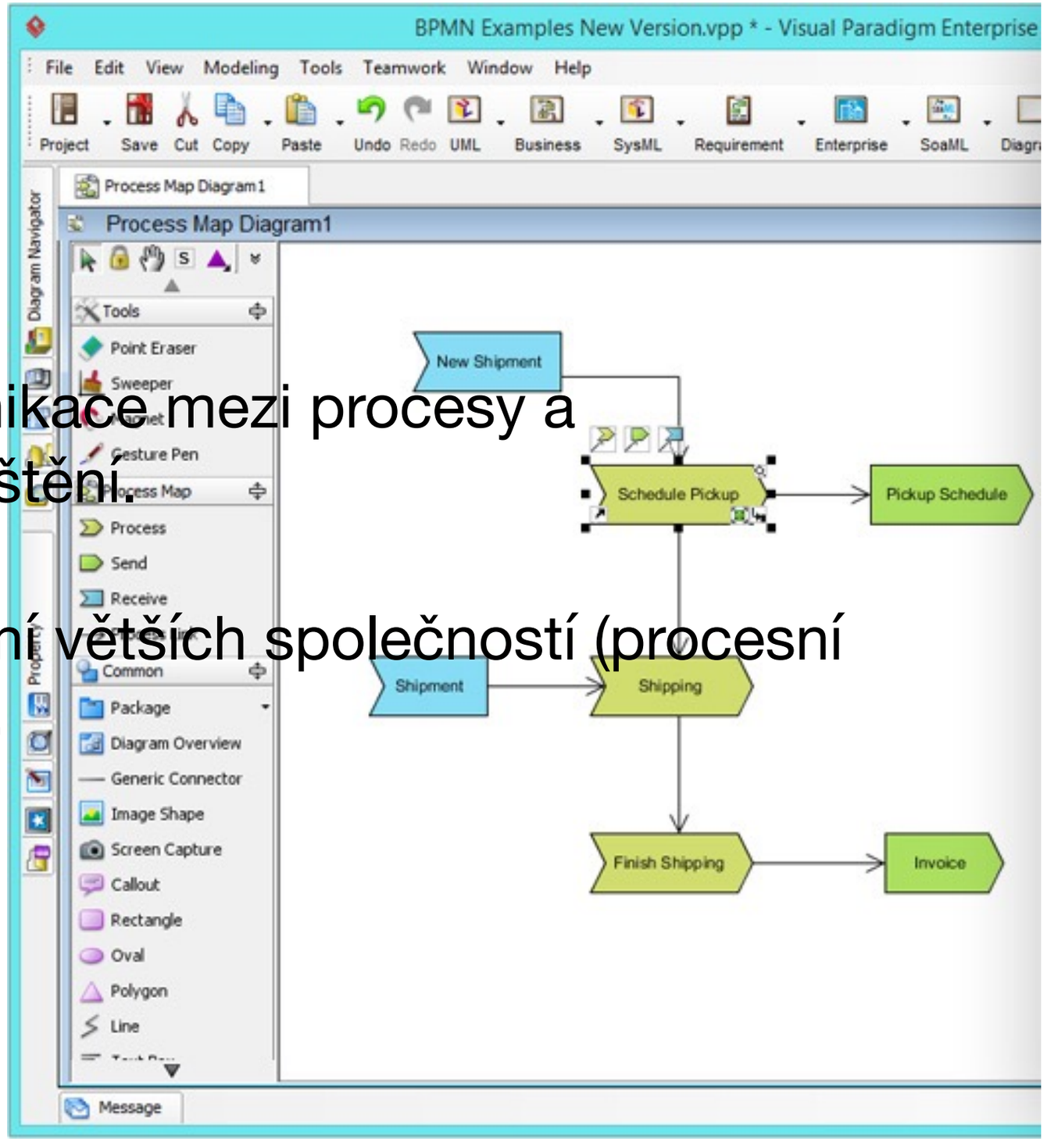
- Jak “cestují” data skrze systém.
- Většinou se dělí na úrovně.
- Ostatní systémy jsou označeny entitou Terminátor.



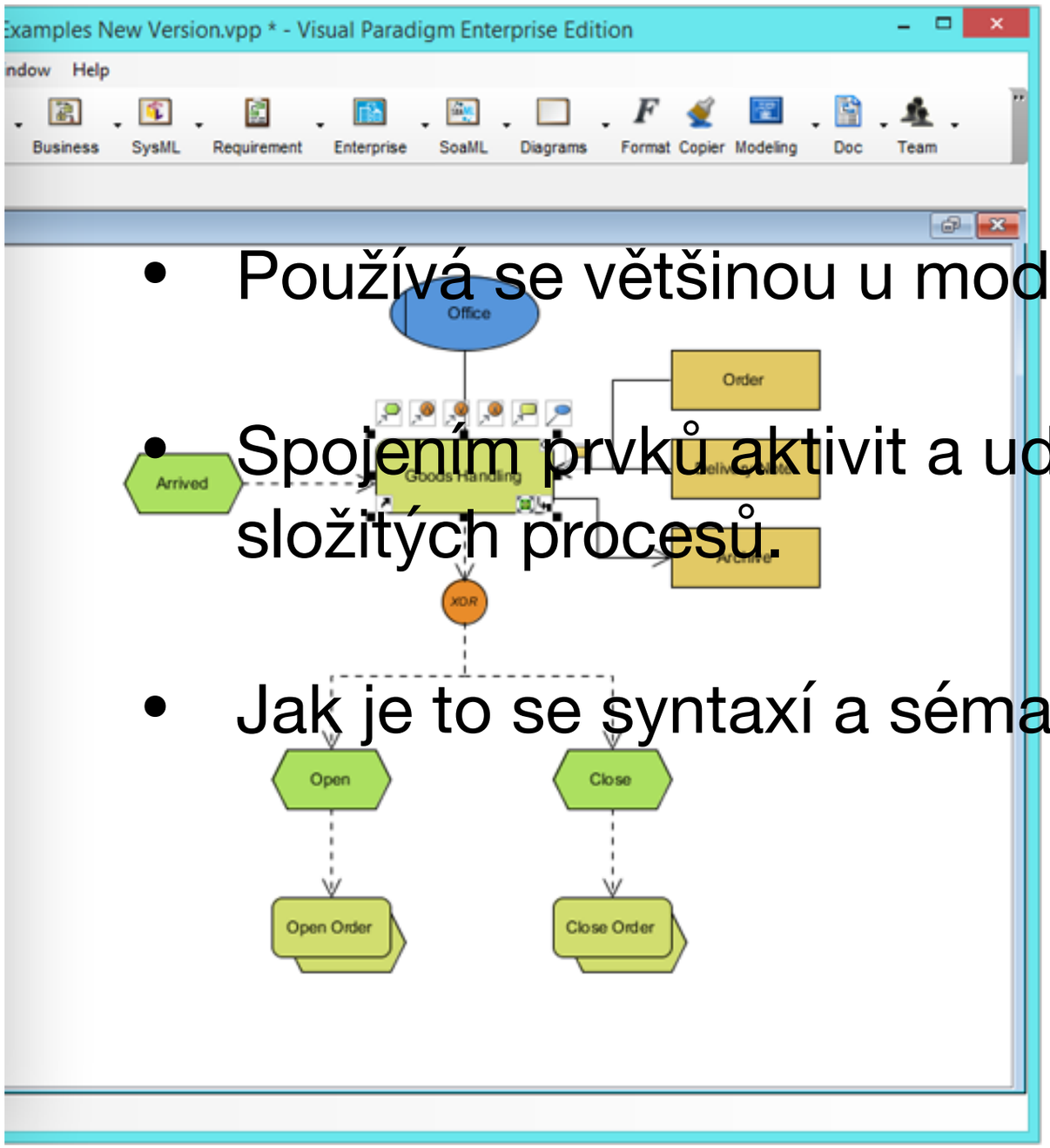
Procesní mapy

- Zachycuje způsob komunikace mezi procesy a mechanismus jejich spouštění.

- Používá se pro modelování větších společností (procesní diagram by nám nestačil).

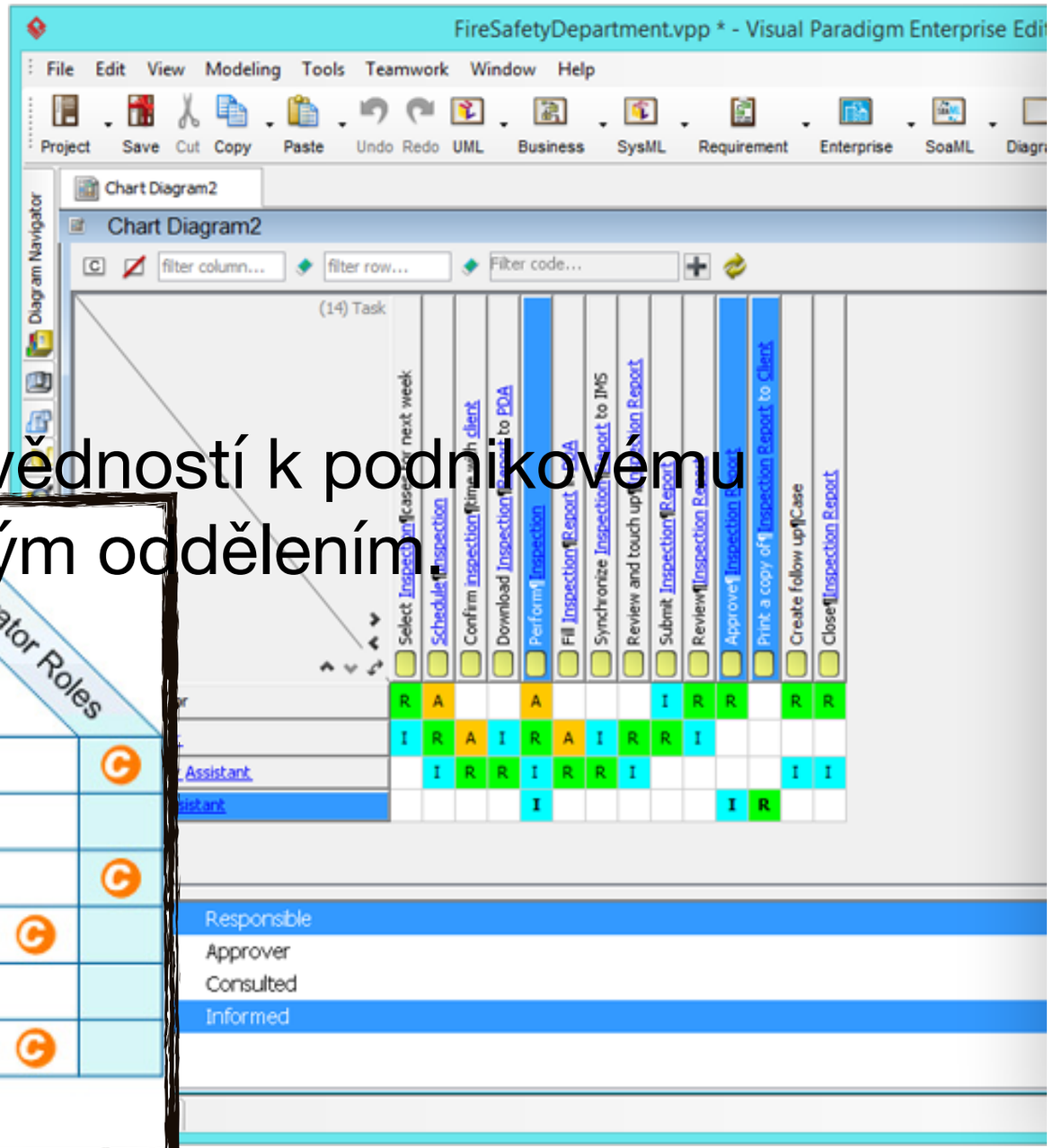


Event-Driven Process chain



- Používá se většinou u modelování ERP systémů.
- Spojením prvků aktivit a událostí dovoluje tvorbu i velmi složitých procesů.
- Jak je to se syntaxí a sémantikou?

Responsibility assignment matrix - RACI chart



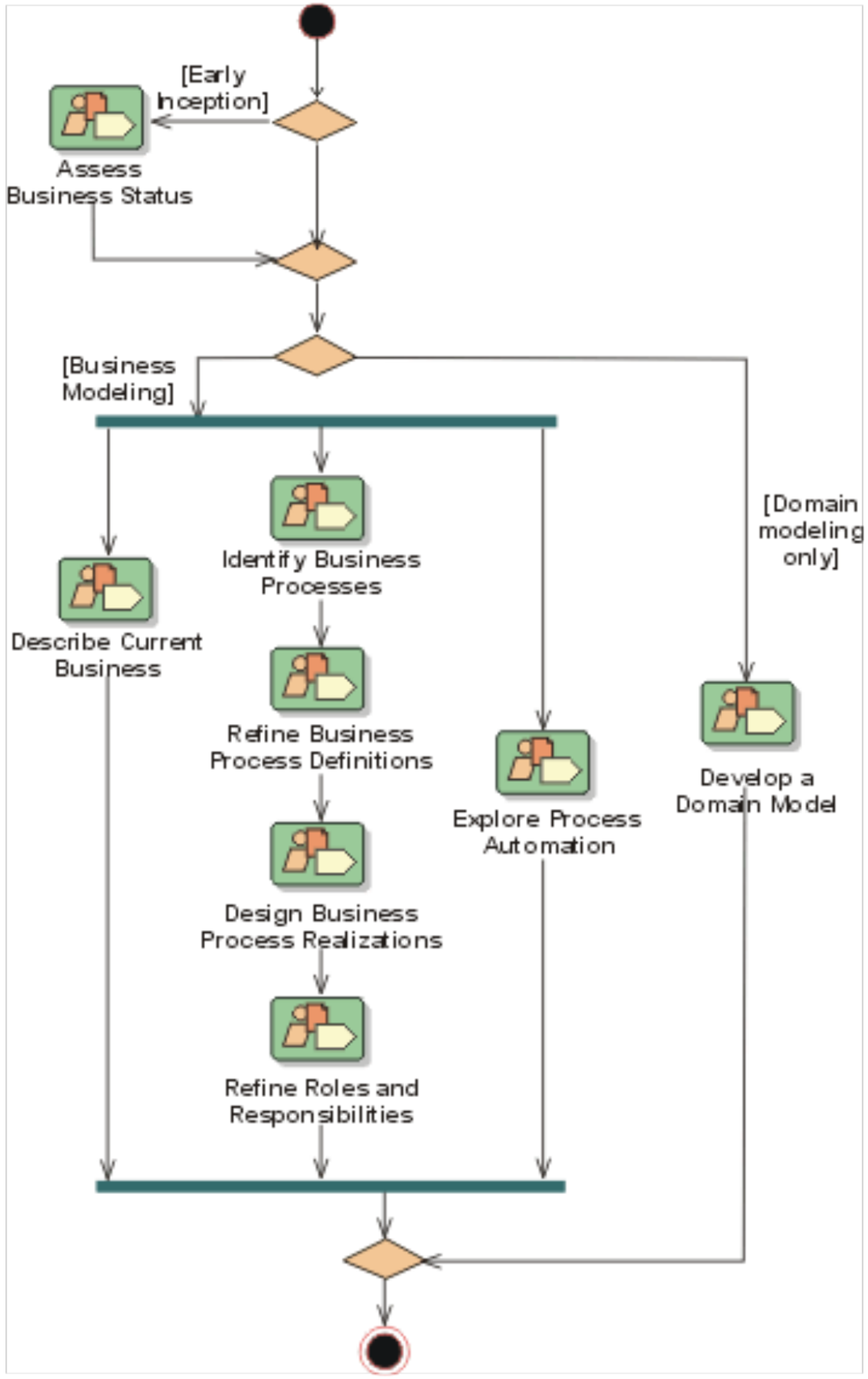
- Pomůže nám při určení odpovědností k podnikovému procesu ve vztahu k jednotlivým oddělením.

Artifact	Product Owner	User	Project Manager	System Analyst	Software Architect	Use Case Specifier	Developer Roles	Tester Roles	Administrator Roles
Vision	A	C	C	W	C		C		C
Use Case Model	A	C	C	W					
Software Architecture Document	A				W		C		C
Use Case Specification	A	C		C	C	W		C	
Use Case Realization					C	C	W		
Component					C		W	C	

W = Write / responsible for C = Contribute / review A = Formally Accept

RUP a Business Modeling

- Pochopení problémů organizace a identifikace oblastí vhodných ke zlepšení.
- Ohodnocení dopadu změny na organizaci.
- Zajištění porozumění potřebám organizace mezi všemi (zákazníci, koncoví uživatelé, vývojáři, …).
- Vyvození požadavků na softwarový systém podporující potřeby organizace.

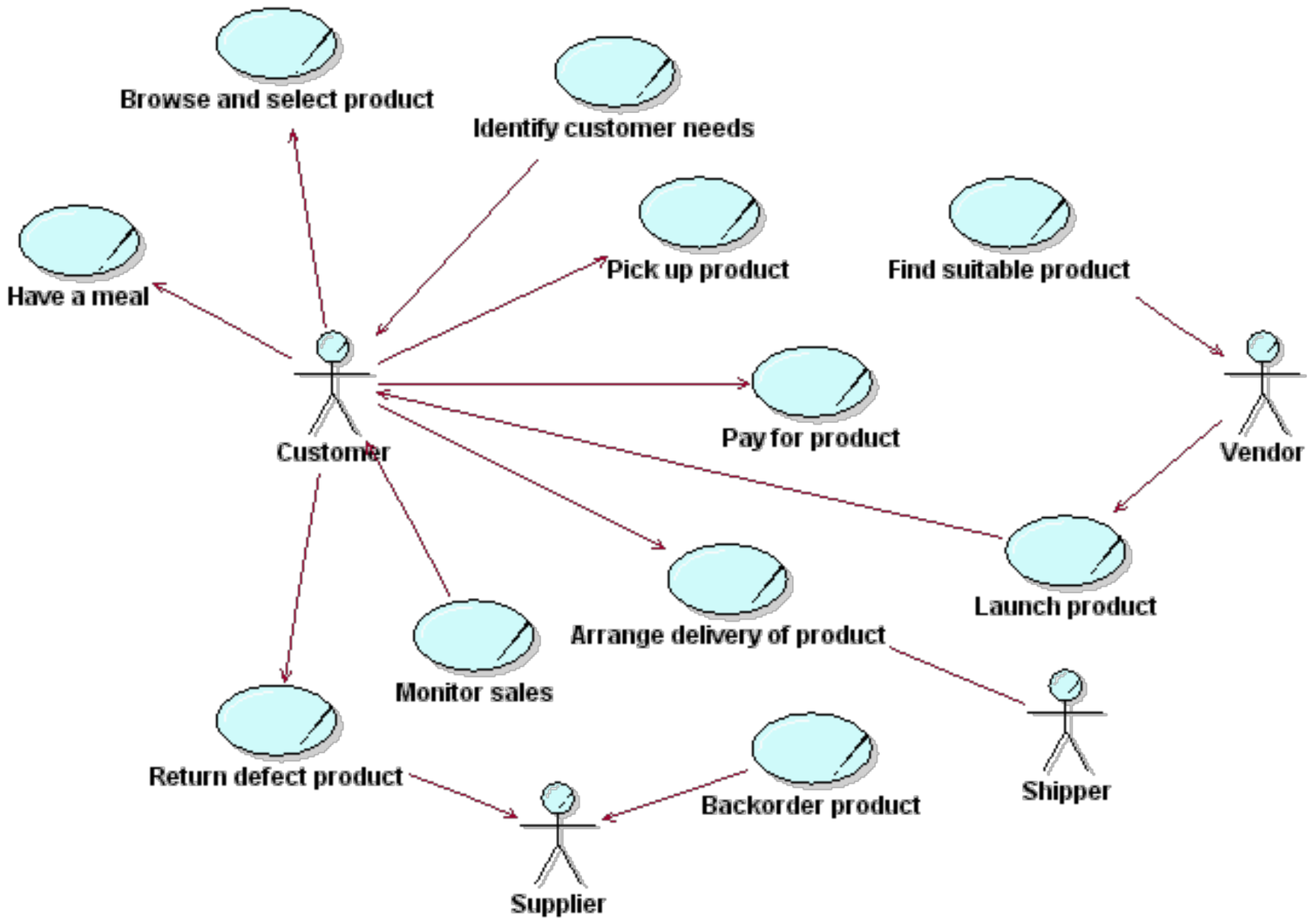

[Early
Inception]
Assess
Business Status
[Business
Modeling]
[Domain
modeling
only]
Identify Business
Processes
Describe Current
Business
Refine Business
Process Definitions
Explore Process
Automation
Develop a
Domain Model
Design Business
Process Realizations
Refine Roles and
Responsibilities

- Artefakty:
    - vize podniku, slovník pojmů, podniková pravidla, procesní modely

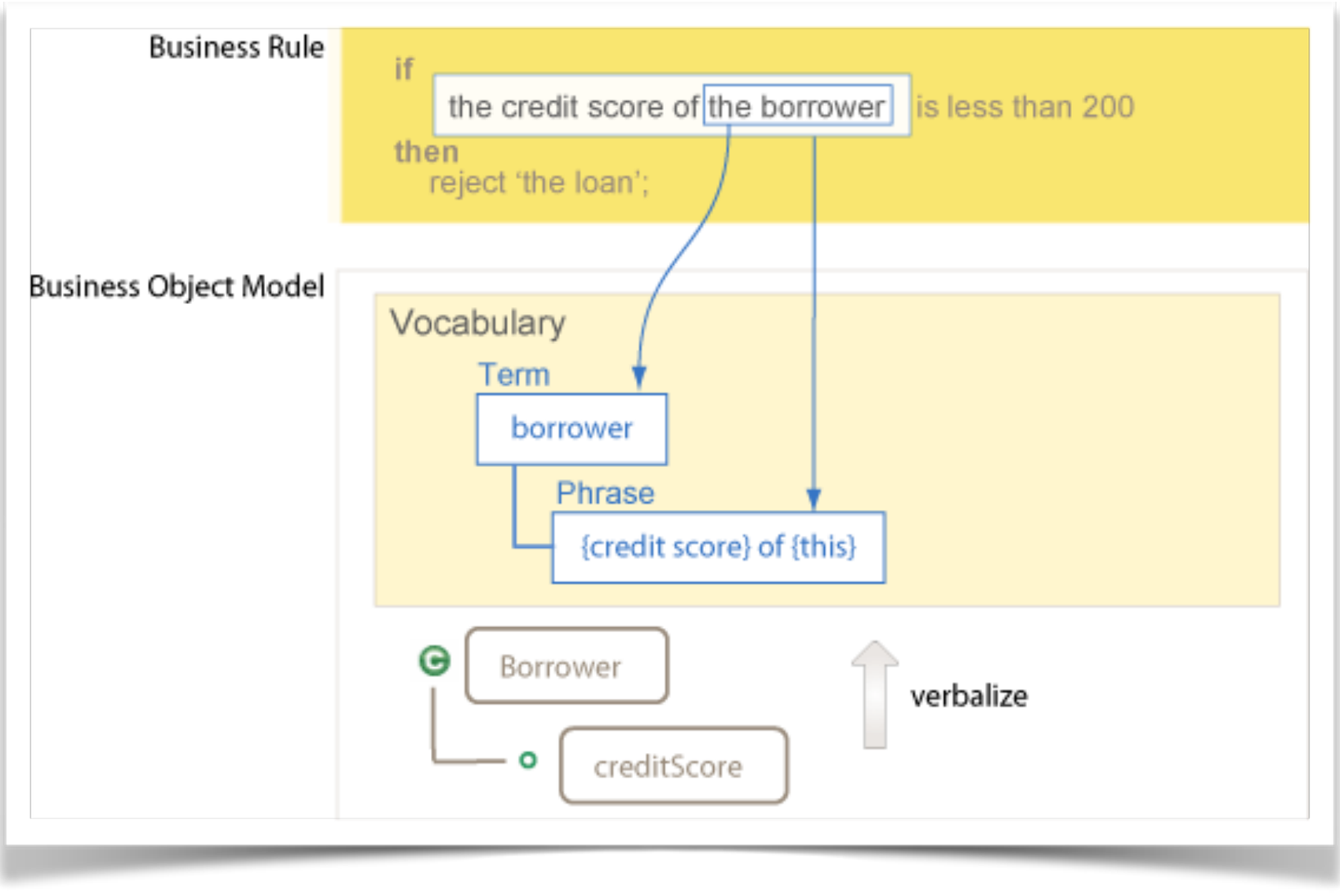
Business Use Case

- Struktura takového scénáře je téměř stejná jako u Use Case:
  - Popisuje jak zaměstnanci pracují (např. zaevidování objednávky, vyřízení obvyklého případu, objednání zájezdu), ale ne za použití aplikace.
  - Textový popis vyřízení daného scénáře.
  - Byznys cíle.
  - Popis vlastního Use Case, práce zaměstnance.
  - Byznys aktoři, vazby v modelu.

Business Use Case



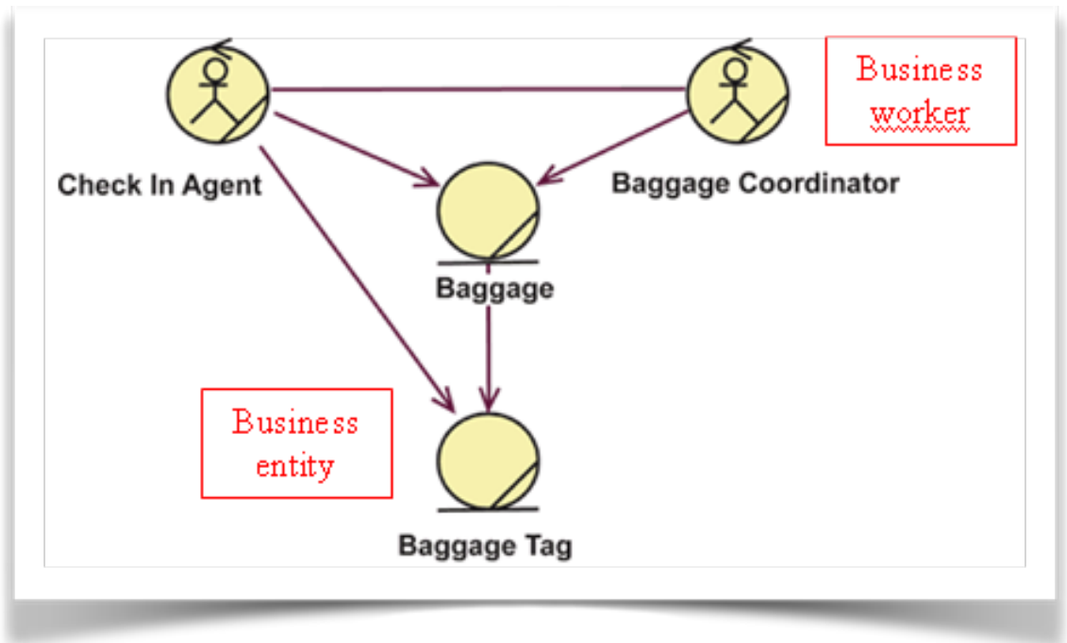
Business Object Model



BOM umožňuje pracovat s business pravidly přívětivou formou

Business Object Model

- Business Use Case model – CO zákazník (uživatel) provádí.
- Business Object Model – říká, JAK toto provádí.
- Dva typy entit
  - ✓aktivní (Business worker)
  - ✓pasivní (Business entity)



Informační systém jako business actor

- Někdy se dva aktoři kontaktují navzájem přes IS.
- Z pohledu modelovaného byznysu je IS také považován za business aktora.
- Příklad: Jako programátoři používáte IDE nástroje. Pokud nefungují jak si myslíte, tak kontaktujete web výrobce (Eclipse) a hledáte, jestli je chyba už reportována. Business worker “Vývojář” interaguje s business aktorem “Dodavatel WWW serveru”.

Kdy je (ne)vhodné BOM použít?

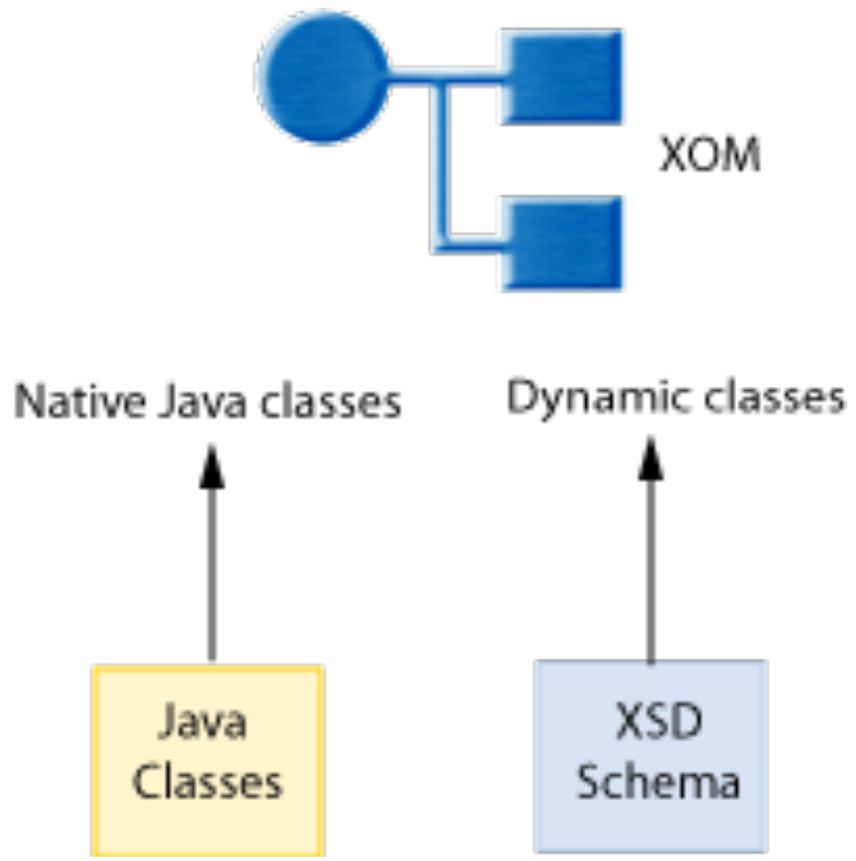
- Pokud začínáte s novým projektem v neznámé nebo dosud dobře neprozkoumané doméně je vhodné BOM použít.

- V projektech zaměřených na funkční architekturu naopak nepomohou.

- Pokud zpracováváme jen malé dávky dat, je lépe se jim vyhnout.

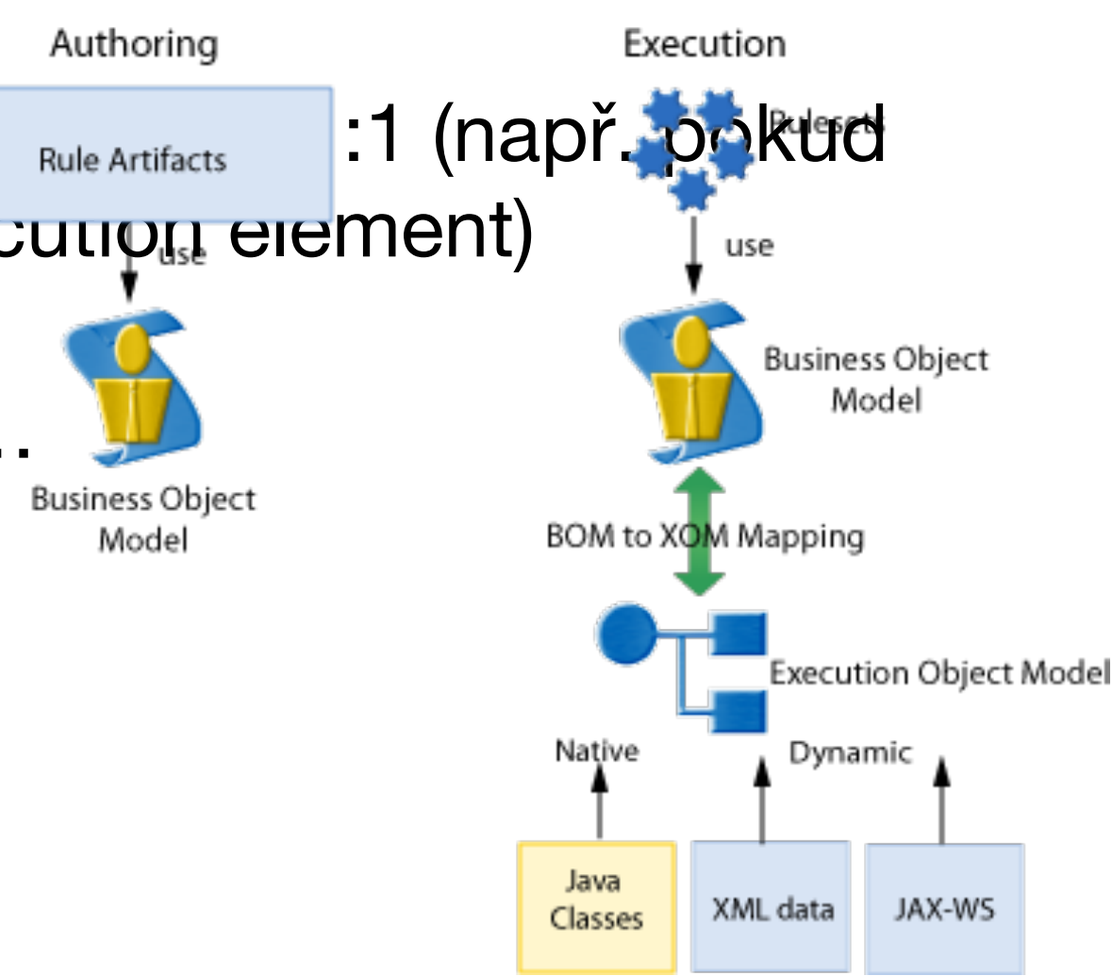
Execution object model (XOM)

- Je model, ve kterém běží business pravidla.
- Skládá se z:
    - Java tříd
    - XML schématu



Z BOM do XOM

- Každý BOM musí mít definovaný XOM.
- Neznamená to ale, že vztah m :1 (např. pokud business element pochází z execution element)
- Lépe je si to ukázat na příkladu…



Business elements vytvořené z XOM

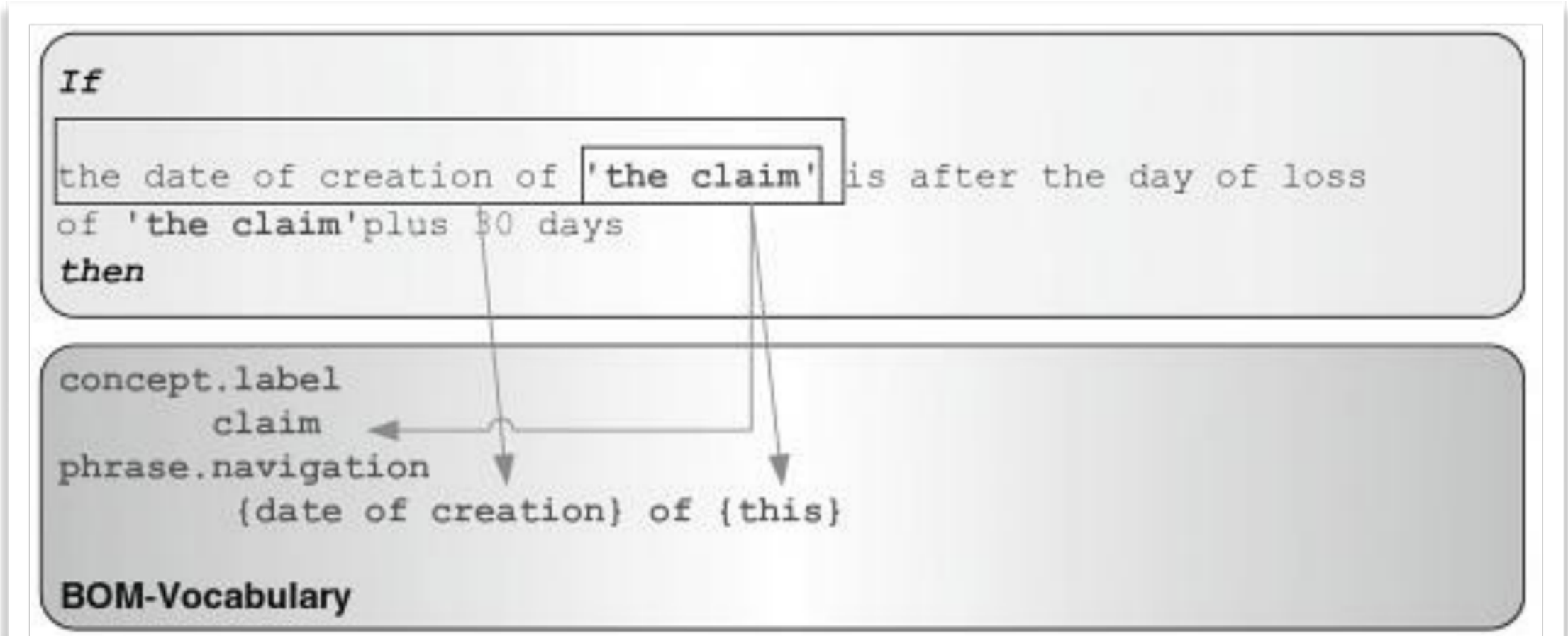
Java element...	**Becomes element in the BOM...**
Nongeneric public class.	Class with the same name.

Class that implements the `java.util.Collection` interface.
For example:
`MyCollection implements java.util.Collection`
	Class with a collection domain to be recognized as a collection when verbalized.
	For example:
	`public class MyCollection implements java.util.Collection {`
	`domain 0, * ;`
	`}`

Public constructor.	Constructor with the same parameters.
Public attribute.	Attribute with the same name and return type.
Final attribute.	Read-only attribute with the same name and return type.

Public static final attributes whose types are the current class.
For example:
`public class Color {`
`private Color(String name) {…}`
`public static final Color red = new Color("red");`
`public static final Color blue = new Color("blue");`
`public static final Color green = new Color("green");`
`}`
	Enumerated domain of static references of the class.
	For example:
	`public class Color {`
	`domain { static red, static blue, static green }`
	`public static final readonly Color red;`
	`public static final readonly Color blue;`
	`public static final readonly Color green;`
	`}`

Mapování BOM-XOM



Mapování BOM-XOM

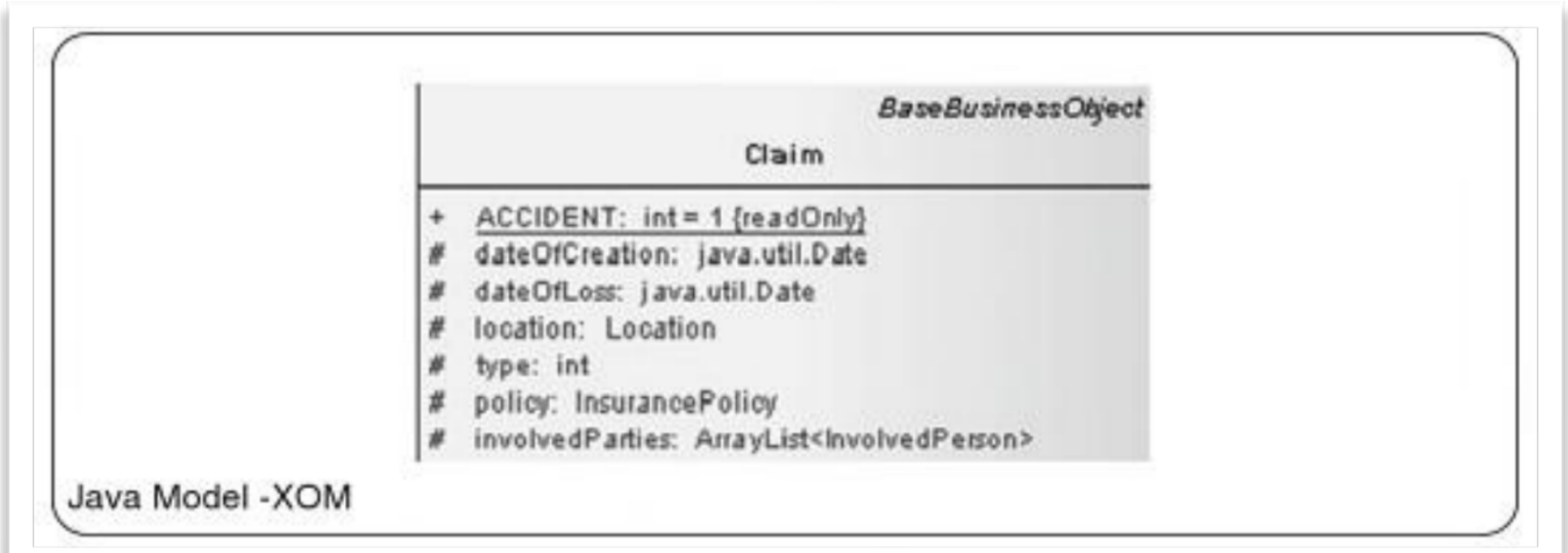
```
package abrd.claim;

public class Claim extends abrd .claim.BaseBusinessObject {
    public readonly java.util.Date dateOfCreation;
    public abrd.claim.Location location;
        …
}
```

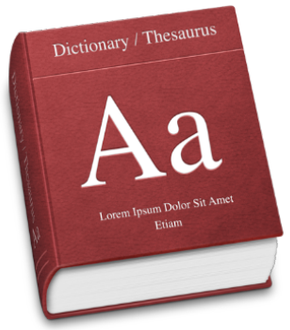
BOM - Data model

```
Java class abrd.claim.Claim -> BOM class abrd.claim.Claim
Java method abrd.claim.Claim.getDateOfCreation() -> BOM
readonly attribute abrd.claim.Claim.dateOfCreation
...
```

BOM to XOM mapping

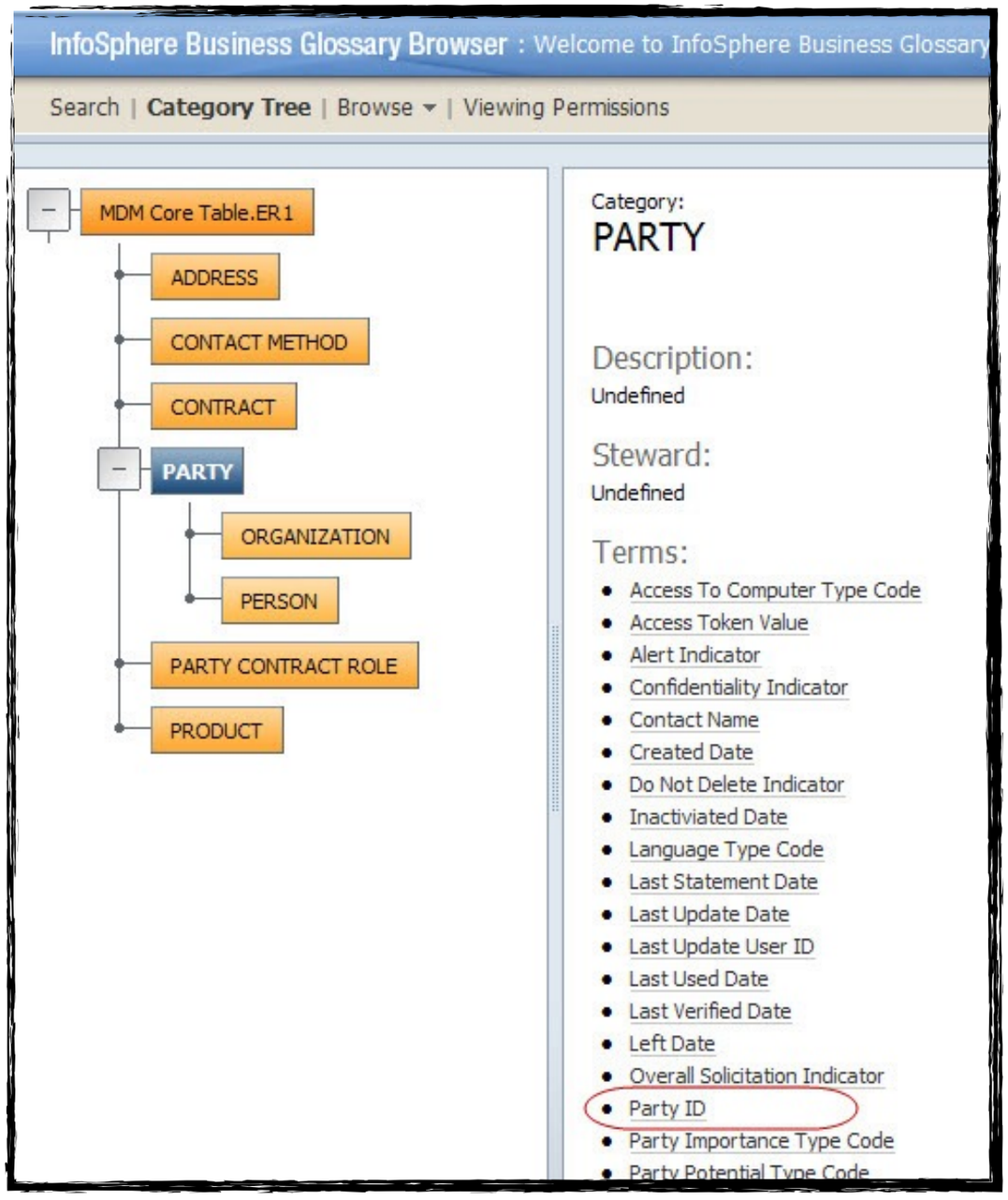
Mapování BOM-XOM





Slovník pojmů

CUNS	See you in school
CX	Cancelled
CYM	Check your E-mail
def	Definitely
DIKU	Do I know you?
DWB	Don't write back
EG	Evil grin
EMA	E-mail address
EML	Email me later
EZ	Easy
F2F	Face-to-face
FWIW	For what it's worth
GAL	Get a life
GALHER (or GALHIM)	Get a load of her (or him)
GBH	Great big hug
GF	Girlfriend
GFI	Go for it
GFN	Gone for now
GG	Good game (or Got to go)
GI	Google it
GL	Good luck (or Get lost)
GM	Good morning (or Good move)
GOI	Get over it
GR8	Great
gratz	Congratulations
GRRRR	growling
GTG	Got to go
GTGP	Got to go pee
GTRM	Going to read mail
h/o	Hold on
HAND	Have a nice day
HB	Hurry back
HHIS	Hanging head in shame
IC	In character (or I see)
IDK	I don't know
IDKY	I don't know you
IMO	In my opinion



Tweety 36,4 tis. | Sledovaní 45 | Sledující 43 mil. | Lajky 23 | Okamžiky 6

Sledovat

Donald J. Trump

@realDonaldTrump

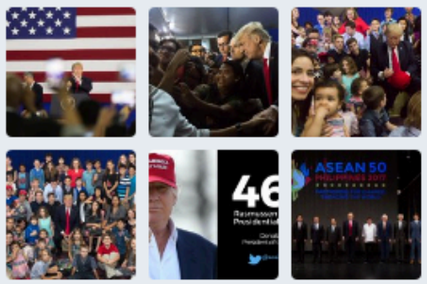
45th President of the United States of America🇺🇸

Washington, DC

Instagram.com/realDonaldTrump

Připojil se březen 2009

2 405 fotek nebo videí



Tweety | Tweety a odpovědi | Média

Donald J. Trump @realDonaldTrump · 6 hod.
Republican Senators are working very hard to get Tax Cuts and Tax Reform approved. Hopefully it will not be long and they do not want to disappoint the American public!

12 tis. 7,7 tis. 36 tis.

Donald J. Trump @realDonaldTrump · 7 hod.
Border Patrol Officer killed at Southern Border, another badly hurt. We will seek out and bring to justice those responsible. We will, and must, build the Wall!

14 tis. 20 tis. 78 tis.

Donald J. Trump @realDonaldTrump · 8 hod.
Big-game trophy decision will be announced next week but will be very hard pressed to change my mind that this horror show in any way helps conservation of Elephants or any other animal.

18 tis. 9,1 tis. 41 tis.

Donald J. Trump @realDonaldTrump · 9 hod.
Shoplifting is a very big deal in China, as it should be (5-10 years in jail), but not to father LaVar. Should have gotten his son out during my next trip to China instead. China told them why they were released. Very ungrateful!

29 tis. 21 tis. 88 tis.

Donald J. Trump @realDonaldTrump · 9 hod.
Odpověď uživateli @realDonaldTrump
Sen. Jeff Flake(y), who is unelectable in the Great State of Arizona (quit race, anemic polls) was caught (purposely) on "mike" saying bad things about your favorite President. He'll be a NO on tax cuts because his political career anyway is "toast."

32 tis. 12 tis. 48 tis.

Poprvé na Twitteru?

Zaregistrujte se a získejte vlastní přizpůsobenou časovou osu.

Zaregistrovat se

Také by se vám mohlo líbit · Aktualizovat

President Trump @POTUS

Hillary Clinton @HillaryClinton

Barack Obama @BarackObama

The White House @WhiteHouse

CNN @CNN

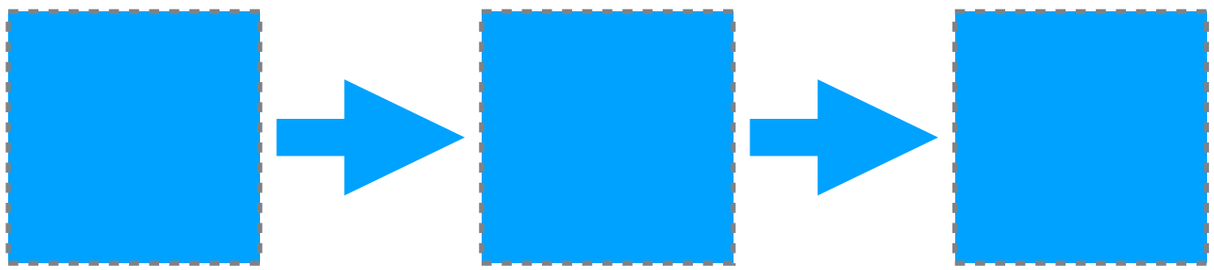
Trendy pro Celosvětově

Charles Manson
Tweety: 150 tis.

#FelizLunes
Tweety: 5 365

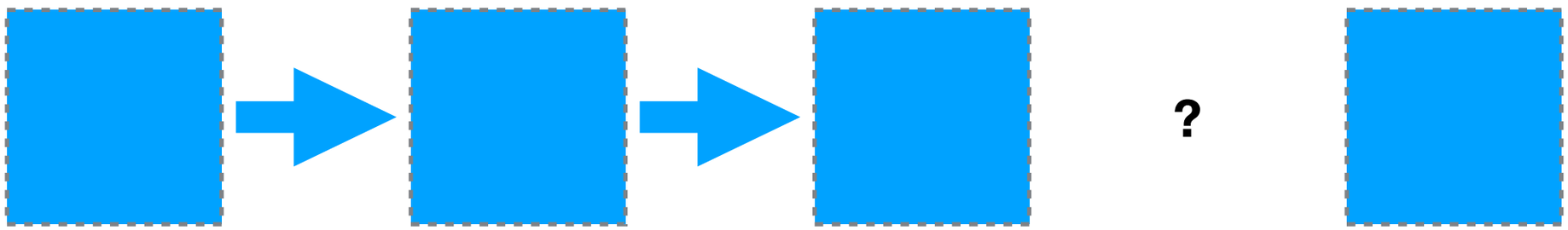
#MasterChefMx
Tweety: 44,3 tis.

Přihlášení

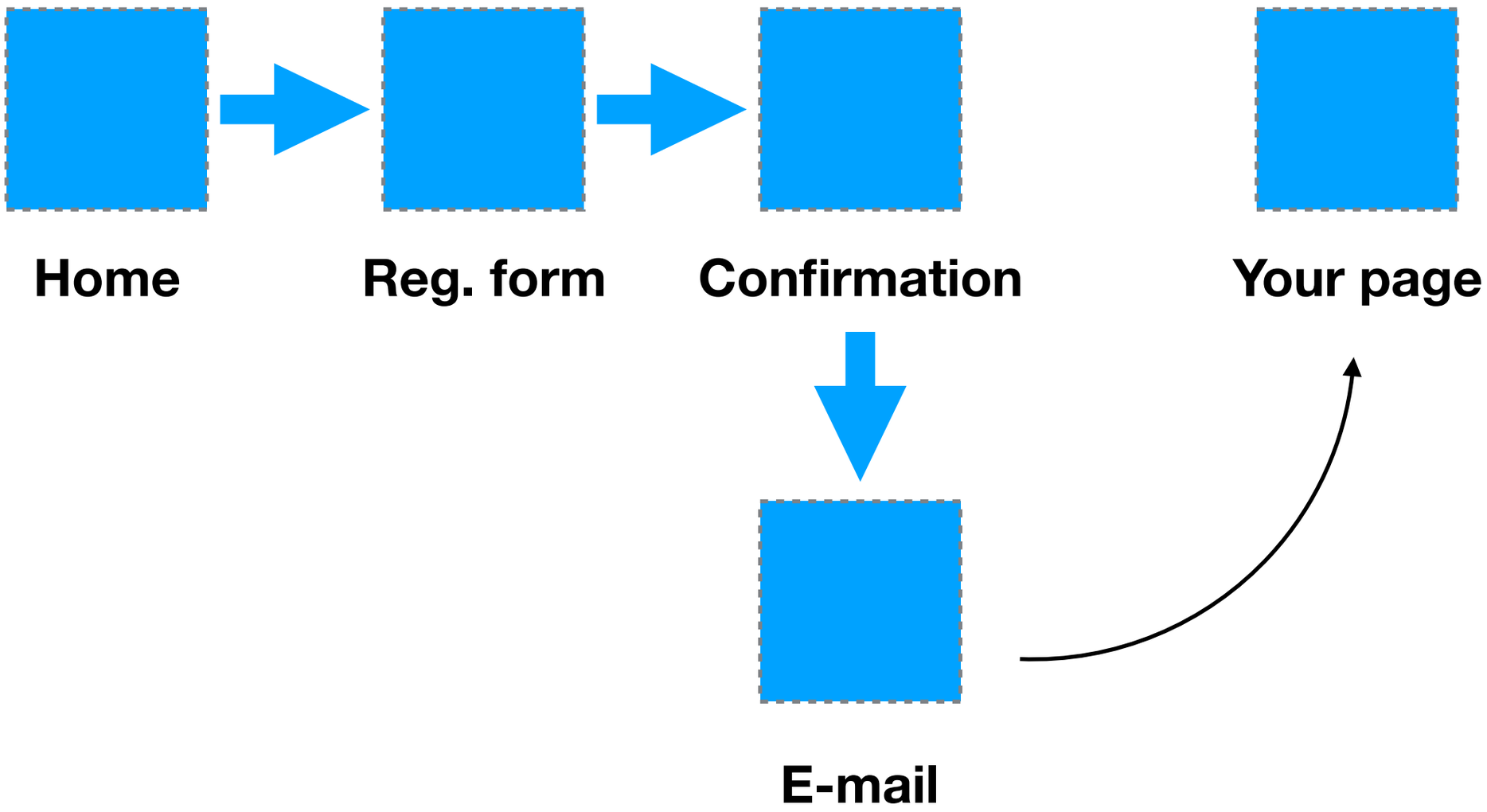


Home **Reg. form** **Confirmation**

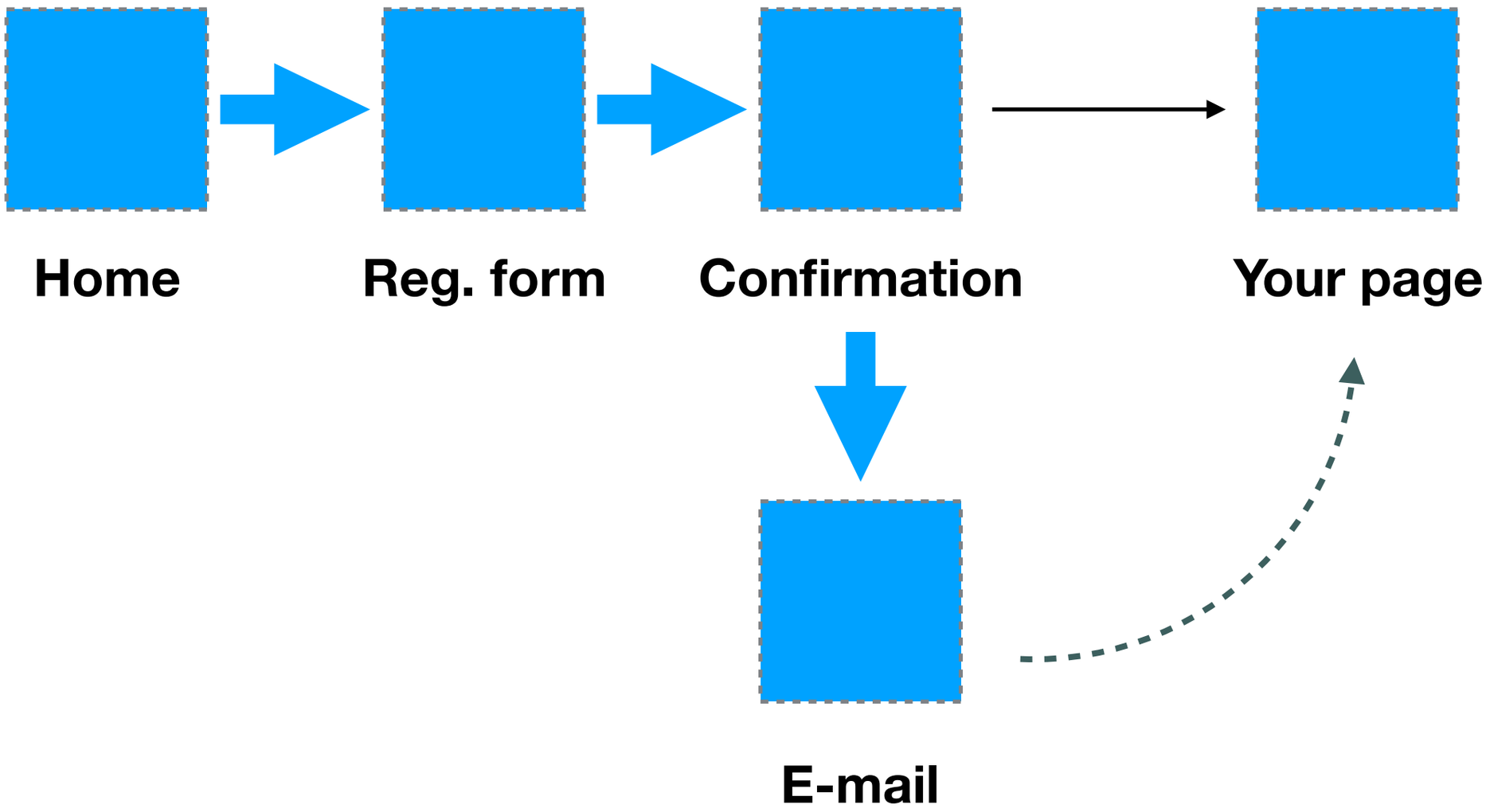
Přihlášení



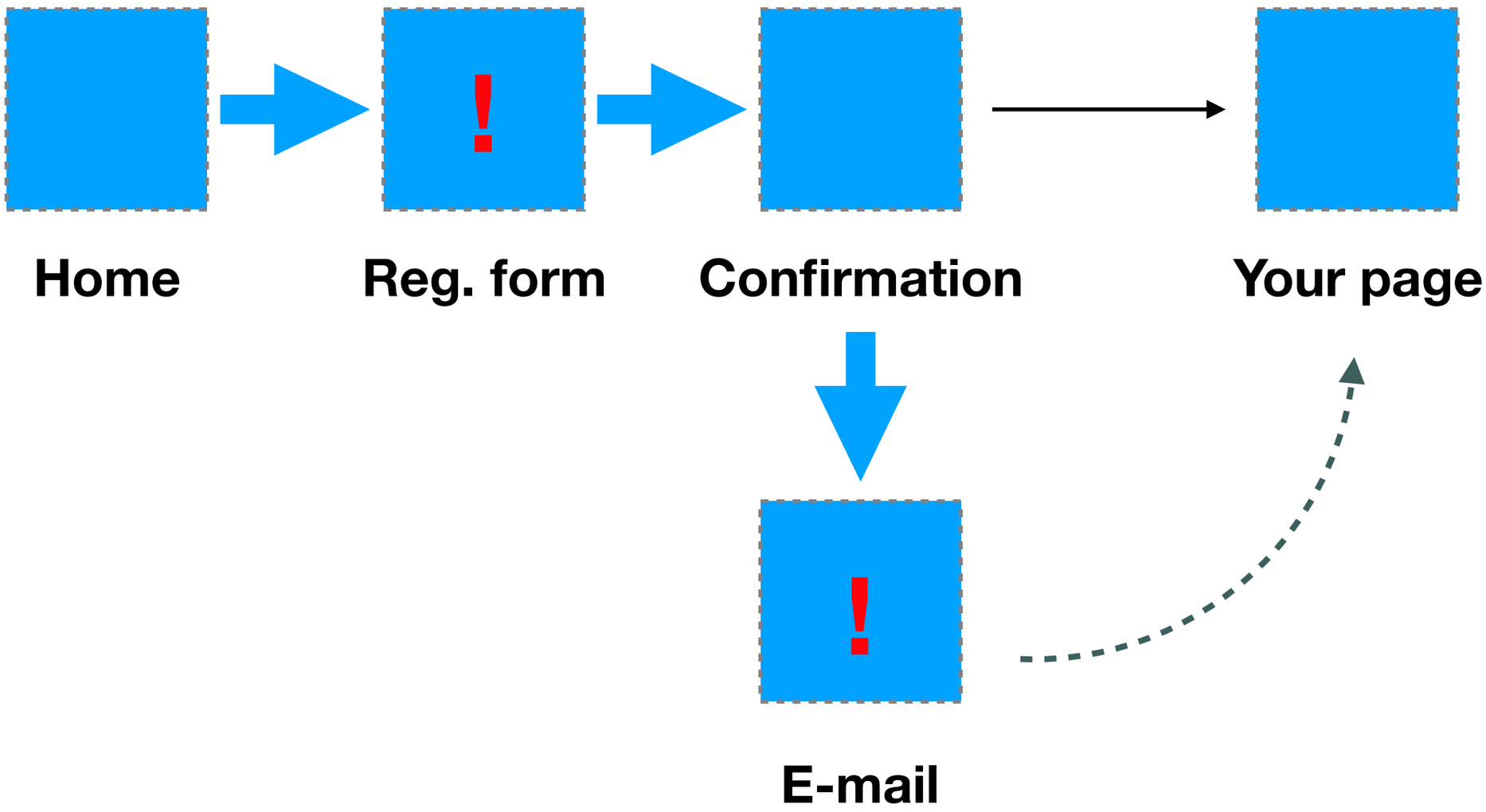
Přihlášení



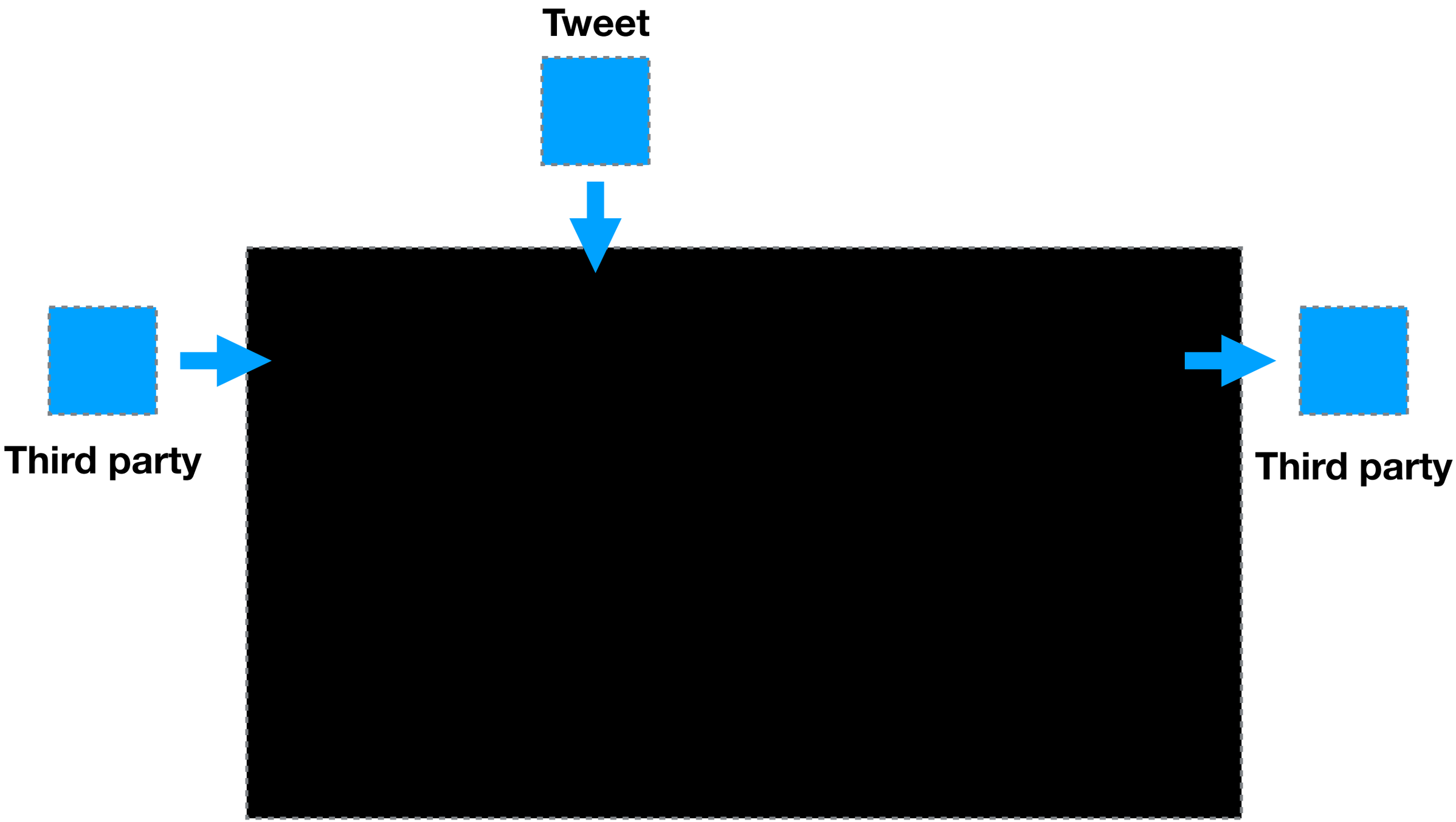
Přihlášení



Přihlášení

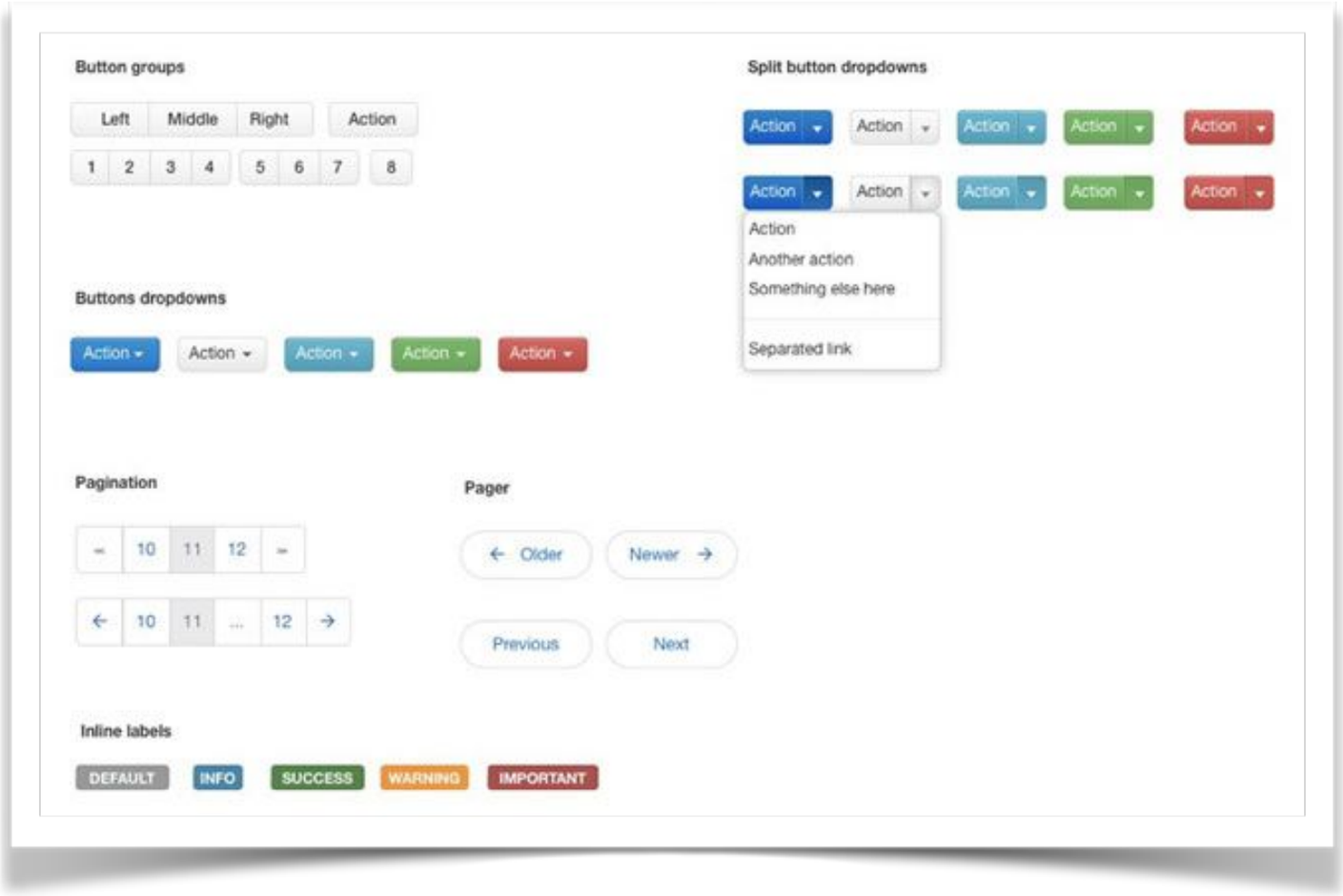


Přihlášení



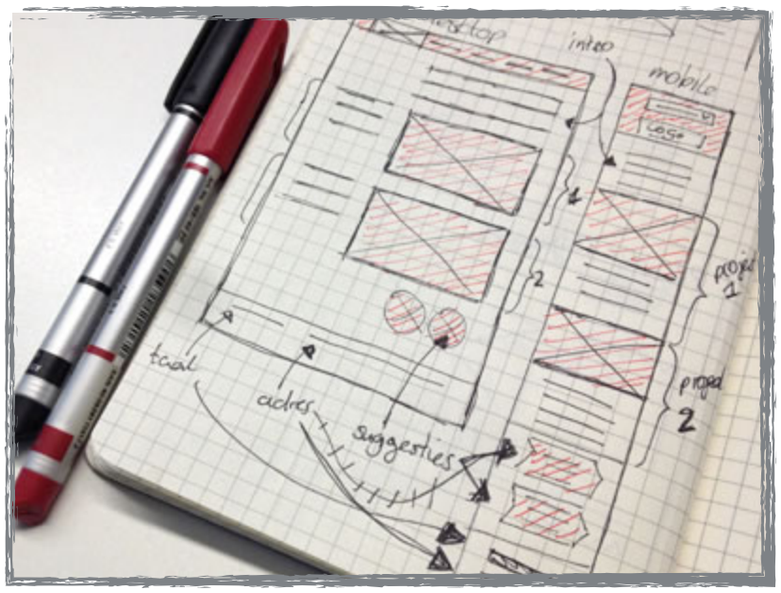
Nástroje

- draw.io
- Twitter bootstrap
- Invision
- Zeplin



Wireframes

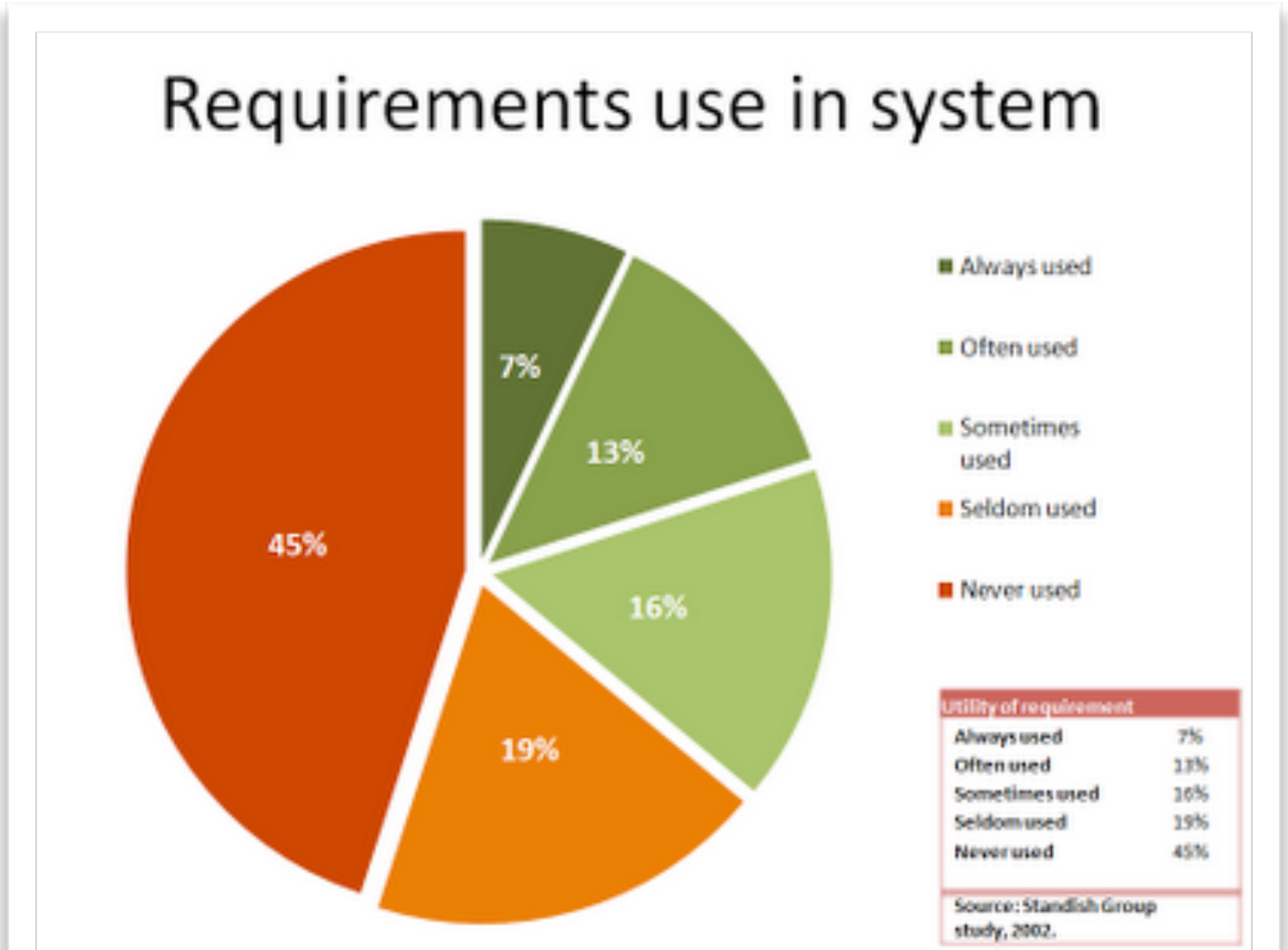
- Výhodné pro upřesnění problémové domény.
- V určitých případech můžou nahradit podnikový proces.
- Vhodné použití u startupových projektů
- Nemusí být vždy v digitální formě…

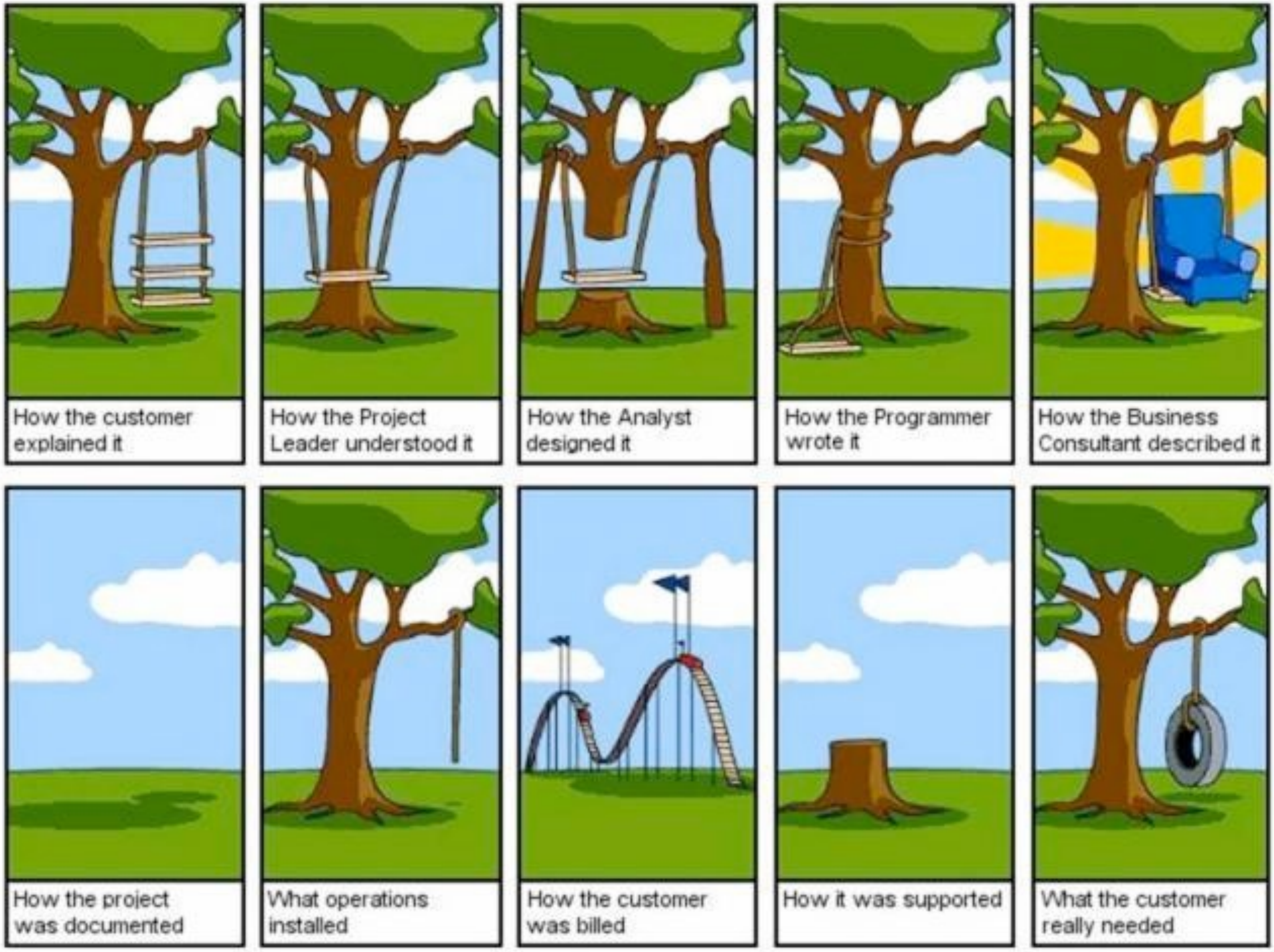


Cvičení

- Dejte dohromady týmy z ROPRu.
- Úkol:
    - Vytvořit IS STAG
    - Jste v roli byznys konzultantů
    - Vytvořte Business Model Canvas
    - Identifikujte základní pojmy z této domény a vytvořte slovník pojmů
    - Snažte se co nejlépe vysvětlit vám známé pojmy, aby to pochopil “běžný” programátor

Requirements




How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Druhy požadavků

- **Funkční** - co má systém dělat, budoucí chování

  ✓Use Case systému – základní způsoby použití, ne detaily

- **Nefunkční** - další vlastnosti systému, které však nejsou funkčnostmi (laicky řečeno nenajdeme je v menu aplikace)

  ✓možnosti přístupu k systému z více míst,

  ✓maximální přístupová doba,

  ✓počet operací za čas,

  ✓implementované standardy, ..

Vzor: popis požadavků v jazyce uživatele

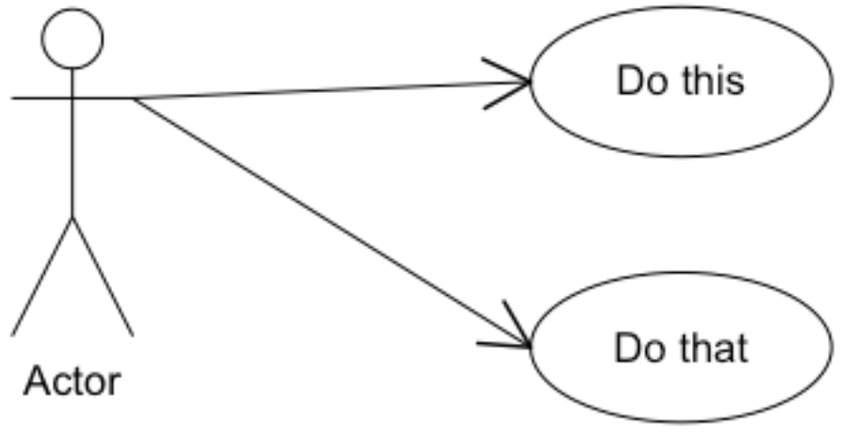
Systém bude dělat to ...
Systém bude umět ono...
Systém bude produkovat ...

Tradiční (dokumentový) způsob

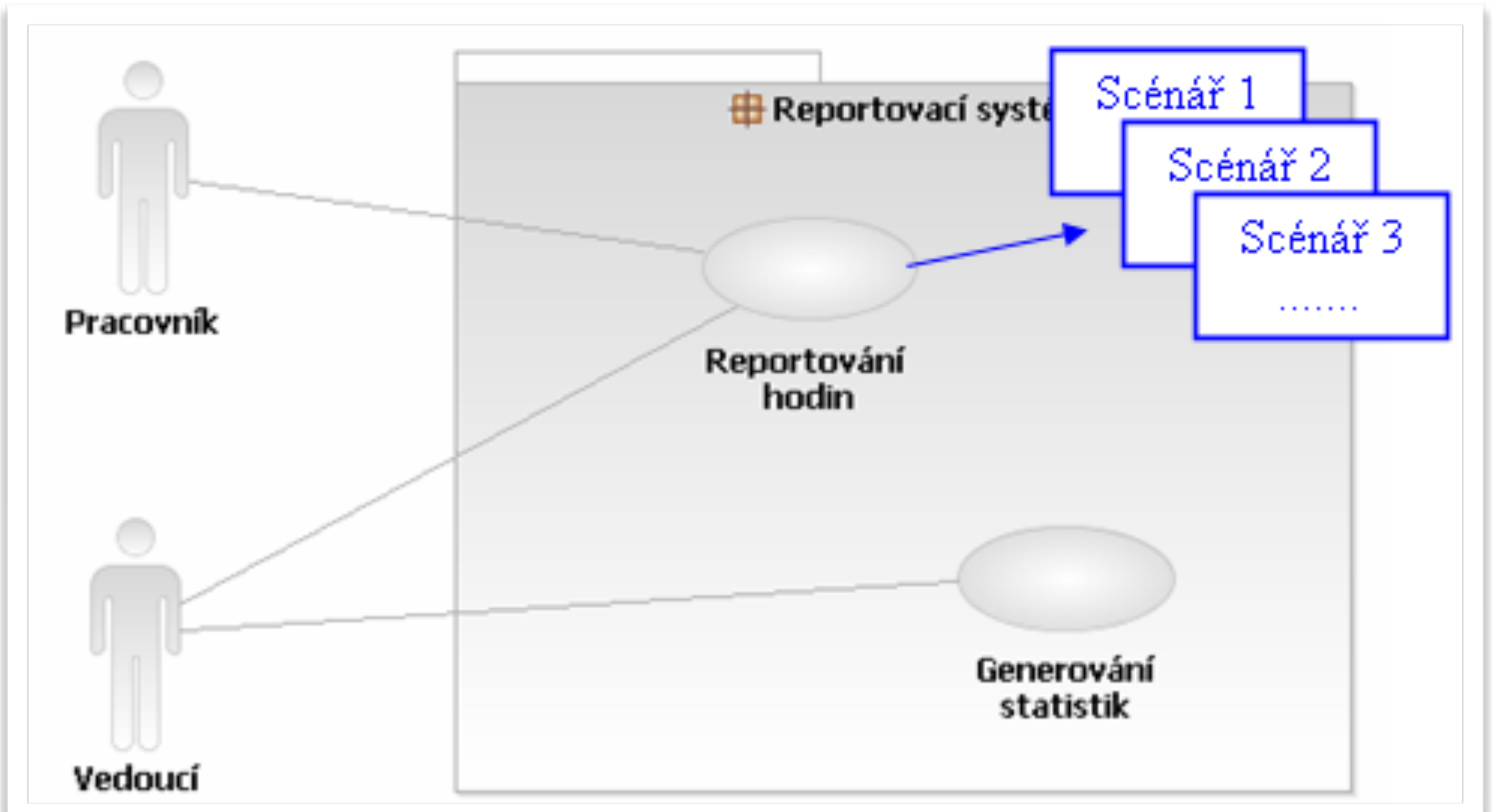
- Detailní dekompozice (funkce, rysy dohromady)
  - Nejasné závislosti
  - Nejasná struktura
- Požadavky popsány co nejvíce detailně

Moderní způsob

- Neexistuje priorita vzhledem k architektuře
  - Požadavky definovány ve formě funkcí, (UC+scénáře)
  - Popis systému z pohledu uživatele
  - Volíme takovou abstrakci jaká je nezbytná, jakou potřebujeme



Use Case a jeho scénáře



Šablona

Introduction:	Short introduction to the specification
Use Case Description:	Brief description conveys the purpose of the Use Case
Pre-condition:	The state that must be present when a use case may start
Flow of Event:	Description of the dialog between actor and system
Basic Flow:	The “Happy day scenario”
Alternative Flows:	Exception and alternatives
Special Requirements:	Nonfunctional requirements specific to the Use Case
Post-condition:	Possible states after at use case has finished
Extension Points:	Definitions of locations of extension points
References:	List of all references

Příklad

Název: Reportování hodin
Aktor: Zaměstnanec
Počáteční podmínka: Zaměstnanec odpracoval určitou dobu na projektu

Tok událostí (basic flow):

1. **Zaměstnanec vybere správu projektů.**
2. **Systém zobrazí správu projektů.**
3. **Zaměstnanec vybere projekt na kterém pracoval.**
4. **Zaměstnanec zadá počet hodin a datum.**
5. **Systém ověří zadaná maximální počet hodin a datum.**
6. **Zaměstnanec odešle zapsaná a ověřená data.**
7. **Systém uloží data.**

Alternativní toky:
4a. …

Chyby při tvorbě UC

- **Příliš mnoho UC a jejich funkční dekompozice** – nepochopení UC, jako UC vystupují jednotlivé scénáře a ne jejich obálka. Ztráta celkového pohledu na daný příběh/ztráta vazeb

- **CRUDL Use Casy a scénáře** – nenásledují kroky, které vykonává uživatel, ale spíše kopírují databázové operace (Create, Read, Update, Delete, List), často produkt programátorů

- **Zahrnutí návrhových rozhodnutí** (klikneme na tlačítko, vybereme z databáze, vyhledávací klíč, seznam s rolovací lištou zobrazený červeně) – nutí nás dělat návrhová rozhodnutí unáhleně, předčasně (mají nedobrý vliv na architekturu a mohou ovlivnit spoustu věcí, resp. způsobit spoustu problémů)

Nefunkční požadavky

Podle IEEE 830 či FURPS+

- použitelnost (usability) – lidský faktor, estetičnost, jednoduchost a intuitivnost aplikace; nejen GUI, ale také dokumentace či tréninkových materiálů;

- spolehlivost (reliability) – týkající se frekvence a dopadu výpadků, jejich předpověditelnosti, obnovy (rychlost, způsob) a přesnosti,

- výkonnost (performance) – mají dopad na funkční požadavky, omezují či specifikují dobu zpracování požadavku, či transakcí, využitelnost paměti pro danou operaci, přístupové rychlosti, přesnost a další,

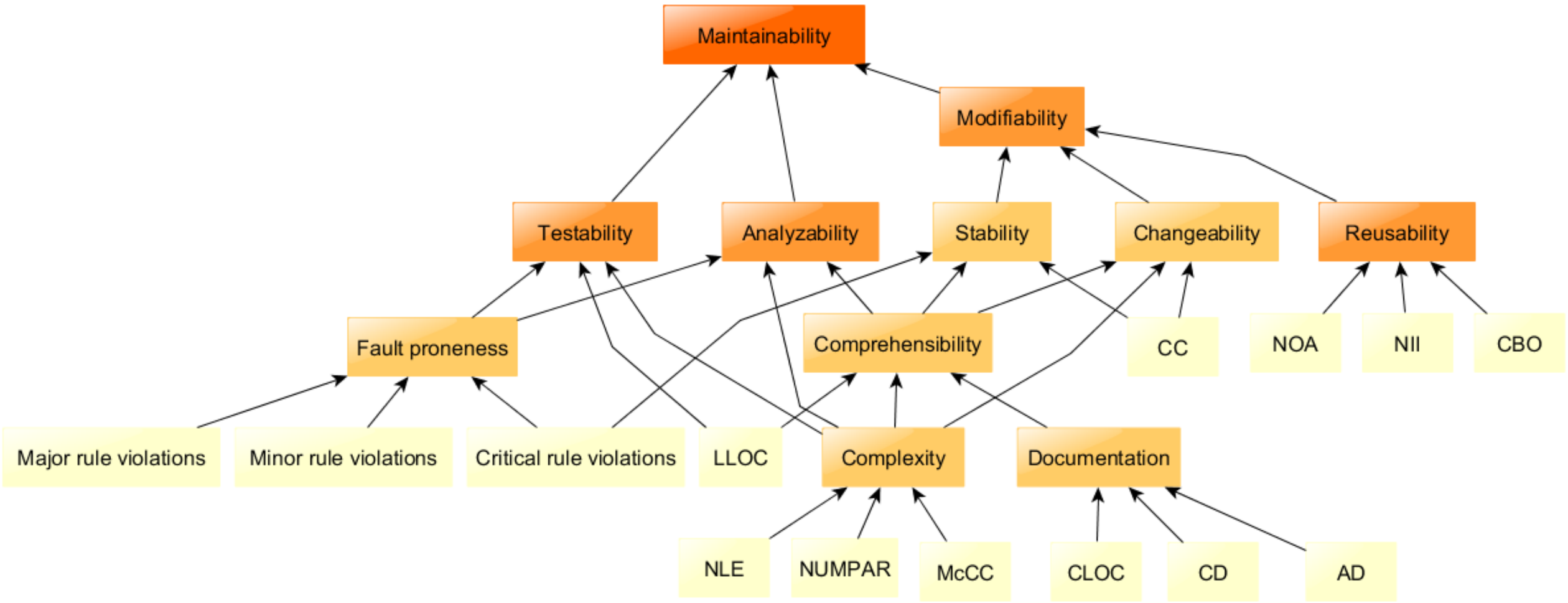
- schopnost podpory (supportability) – udržitelnost aplikace v chodu, jedná se o testovatelnost, rozšiřitelnost atd. Netýkají se pouze požadavků či systému jako takového, ale také procesů a standardů kolem této aplikace!

- Bezpečnost (security) – bezpečné uložení dat, přístup k aplikacím či jejich některým funkcím: zabezpečený vs. nezabezpečený přístup, zobrazení pouze relevantních informací či dokumentů apod.

Nefunkční požadavky obecně

- Zálohování
- Kapacita/propustnost
- Cena - základní, údržba
- Datová integrita
- Deployment
- Failover, disaster recovery
- Dokumentace
- Síťová topologie
- Začlenění open source
- Performance
- Přenositelnost
- Resilience
- Škálovatelnost
- Stabilita

SQuaRE (Software product Quality Requirements and Evaluation)



SEI QAM

Stimul (Stimulus)	Podmínka, která ovlivňuje systém
Zdroj (Source)	Entita, která generuje stimul.
Prostředí (Environment)	Podmínky, za kterých se daný stimul objeví.
Artefakt (Artefact)	Artefakt, který je stimulován.
Odezva (Response)	Odezva, kterou dostaneme jako odpověď na daný stimul.
Míra pro měření odezvy (Response measurem)	Míra, již je daná systémová odezva měřena, ohodnocena.

Příklad

Scenario Refinement for Scenario N		
Scenario(s):		When a garage door opener senses an object in the door's path, it stops the door in less than one millisecond.
Business Goals:		safest system; feature-rich product
Relevant Quality Attributes:		safety, performance
Scenario Components	**Stimulus:**	An object is in the path of a garage door.
	Stimulus Source:	object external to system, such as a bicycle
	Environment:	The garage door is in the process of closing.
	Artifact (If Known):	system's motion sensor, motion-control software component
	Response:	The garage door stops moving.
	Response Measure:	one millisecond
Questions:		How large must an object be before it is detected by the system's sensor?
Issues:		May need to train installers to prevent malfunctions and avoid potential legal issues.

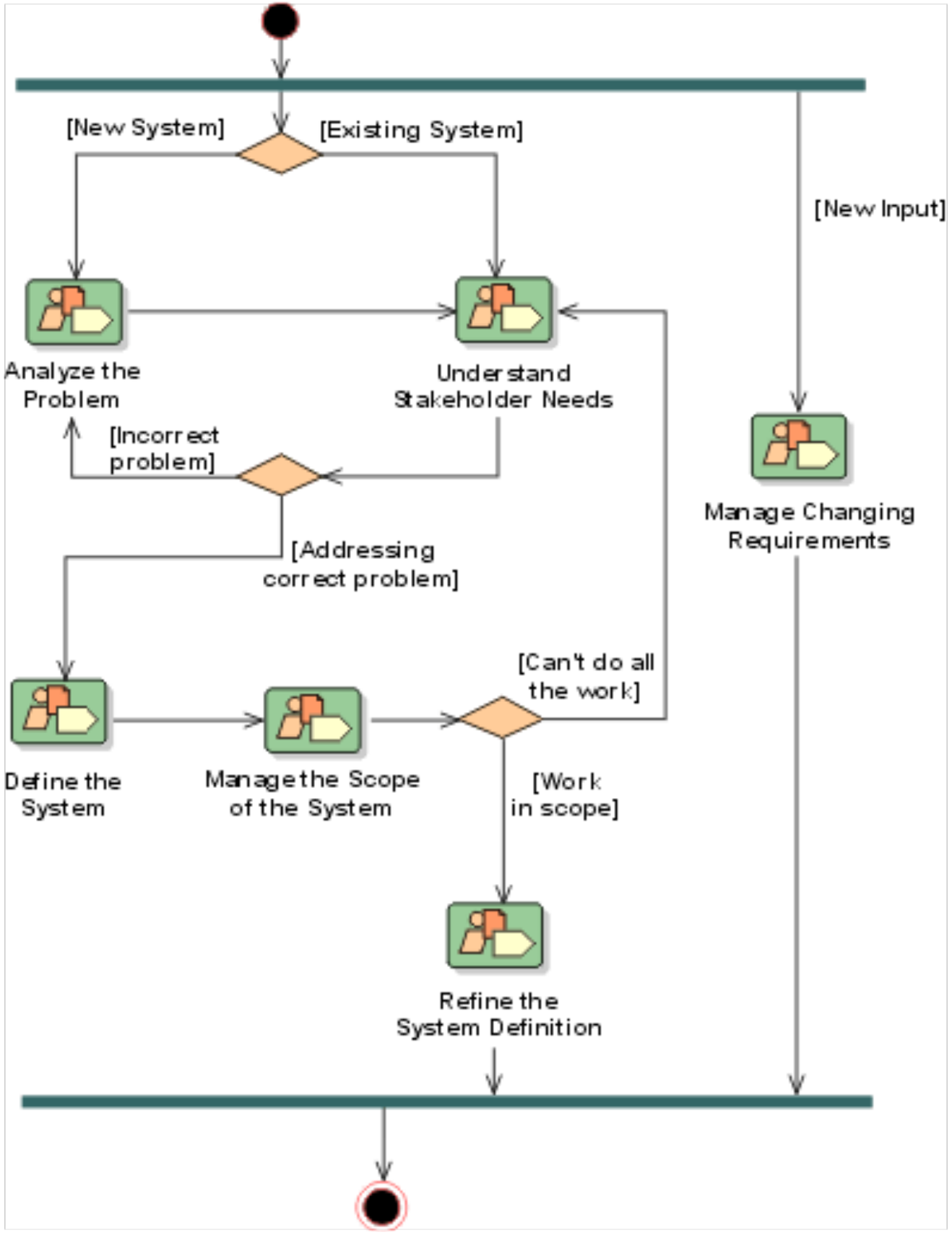
INVEST model kvality

- I - Independent
- N - Negotiable
- V - Valuable
- E - Estimable
- S - Small
- T - Testable

Requirements jako disciplína RUP

- Vytvoření a udržování dohody mezi zákazníkem a dalšími účastníky projektu o tom, co by měl systém dělat.
- Poskytnout vývojářům lepší pochopení požadavků na systém.
- Definovat hranice systému.
- Poskytnout základ pro plánování technické náplně iterací.
- Poskytnout základ pro odhad nákladů a času potřebných k vývoji systému.
- Definovat uživatelské rozhraní systému se zaměřením na potřeby a cíle uživatele.

- RUP je řízen Use Case (Use Case driven)
- Který princip se zde uplatní?


[New System]
[Existing System]
[New Input]
Analyze the Problem
Understand Stakeholder Needs
[Incorrect problem]
Manage Changing Requirements
[Addressing correct problem]
[Can't do all the work]
Define the System
Manage the Scope of the System
[Work in scope]
Refine the System Definition

Aktivity requirements workflow

- Pochopení problému a jeho analýza – porozumění problému v problémové doméně a nalezení k němu řešení.

- Identifikace všech se zájmem na projektu (tzv. stakeholders) – různé zájmy, příliš mnoho stakeholderů s rozdílnými zájmy může způsobit neúspěch projektu.

- Definice systému (scope) a jeho hranic (boundaries) – společně se stakeholdery, co ještě bude systém za problém řešit a co už ne, jaké budou jeho přibližné rysy (features).

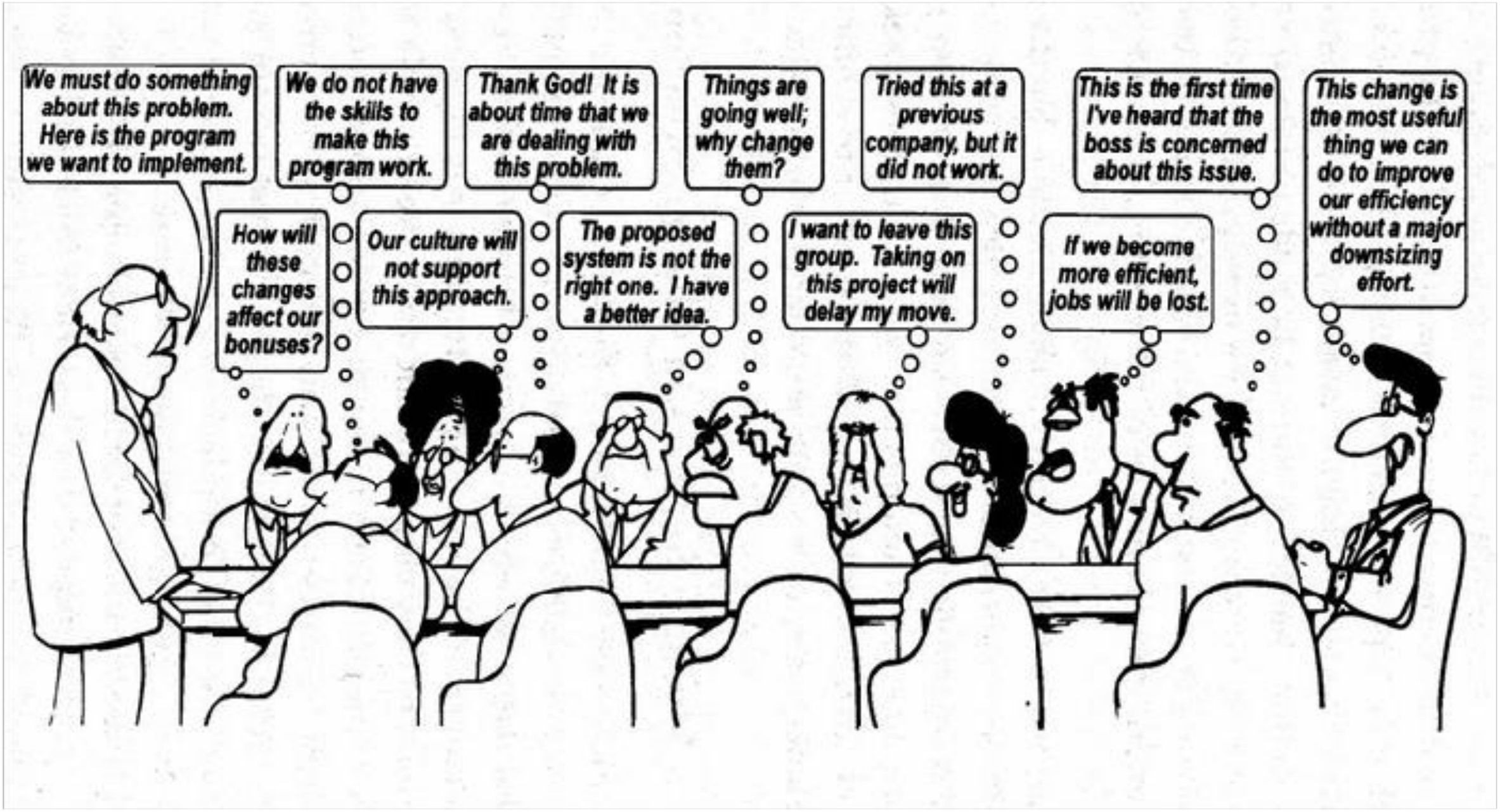
- Identifikace omezení, které musí systém mít (technologie, vazby na systémy, bezpečnostní požadavky, apod)

- Správa měnících se požadavků – dopad a hodnocení nových požadavků a změn (CCB – Change Control Board)

CCB


CUSTOMERS
REGULATIONS
CHANGE CONTROL BOARD
QUALITY ASSURANCE
GLOBAL DEVELOPMENT SITES
Change Management
MANAGEMENT
DOCUMENTATION
TESTING
PRODUCTION
SUPPORT

CCB


We must do something about this problem. Here is the program we want to implement.
We do not have the skills to make this program work.
Thank God! It is about time that we are dealing with this problem.
Things are going well; why change them?
Tried this at a previous company, but it did not work.
This is the first time I've heard that the boss is concerned about this issue.
This change is the most useful thing we can do to improve our efficiency without a major downsizing effort.
How will these changes affect our bonuses?
Our culture will not support this approach.
The proposed system is not the right one. I have a better idea.
I want to leave this group. Taking on this project will delay my move.
If we become more efficient, jobs will be lost.

Identifikace požadavků

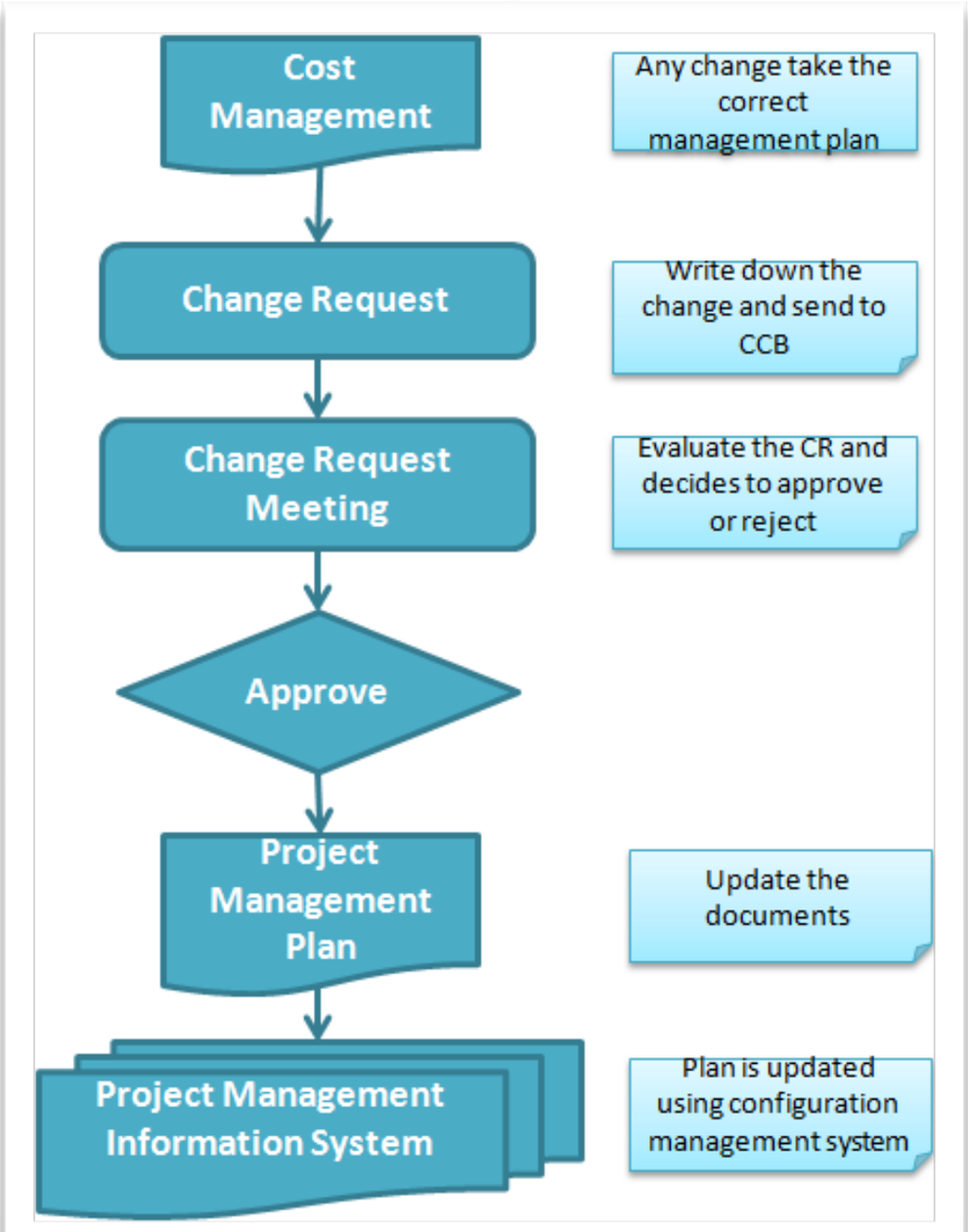
- **Rozhovory s vybranými uživateli –** může být ve formě připravených otázek, které přesně dodržujeme nebo ve formě volného rozhovoru.

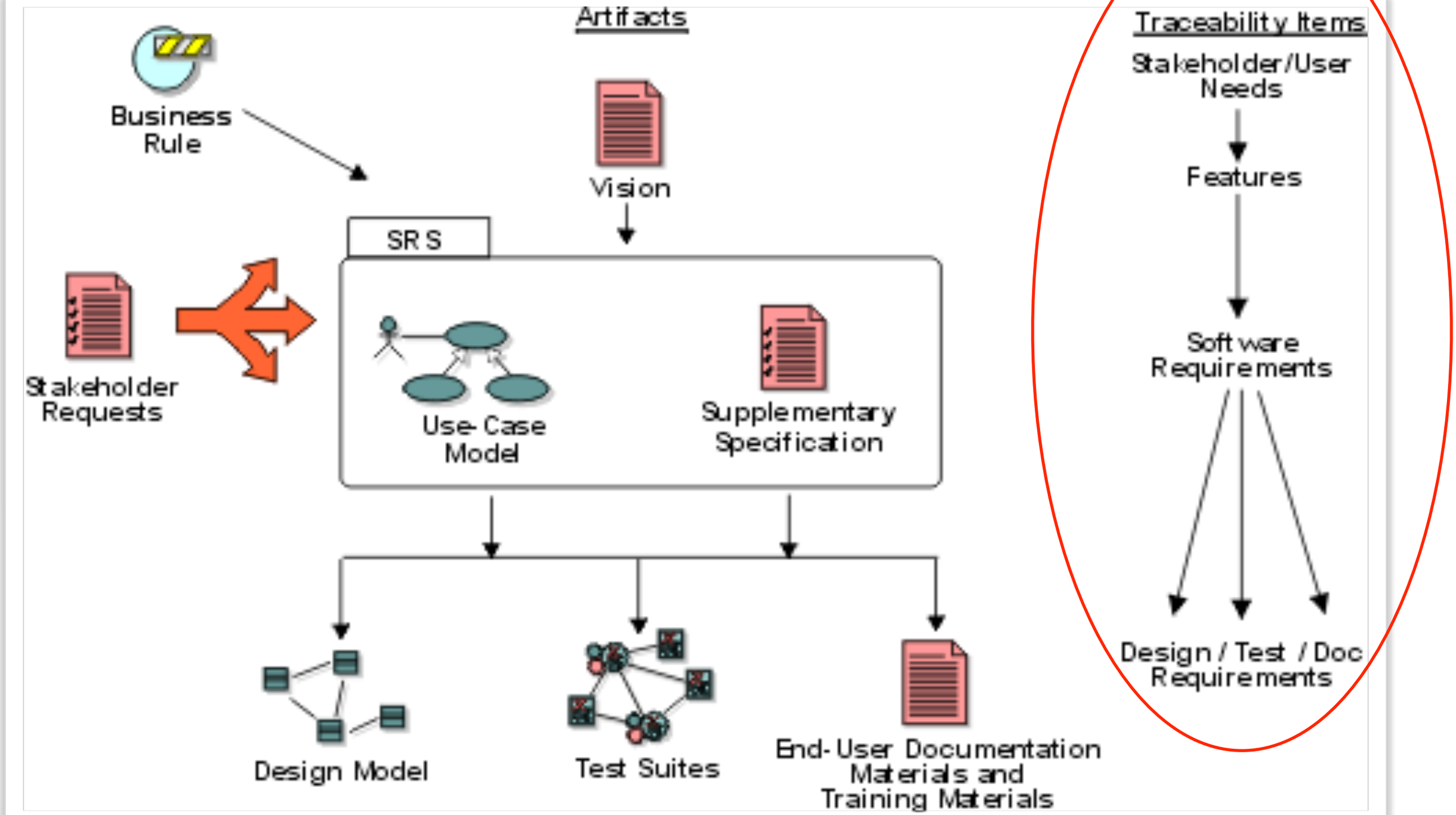
- **Requirements workshop** – jedná se o časově omezenou schůzku, kdy například formou brainstormingu generujeme možné požadavky. Tento workshop je řízen byznys analytikem, který vede směr úvah tam, kam potřebuje.

- **Prototyp** – hrubý nástřel rozhraní (HTML, GUI) pro demonstraci,

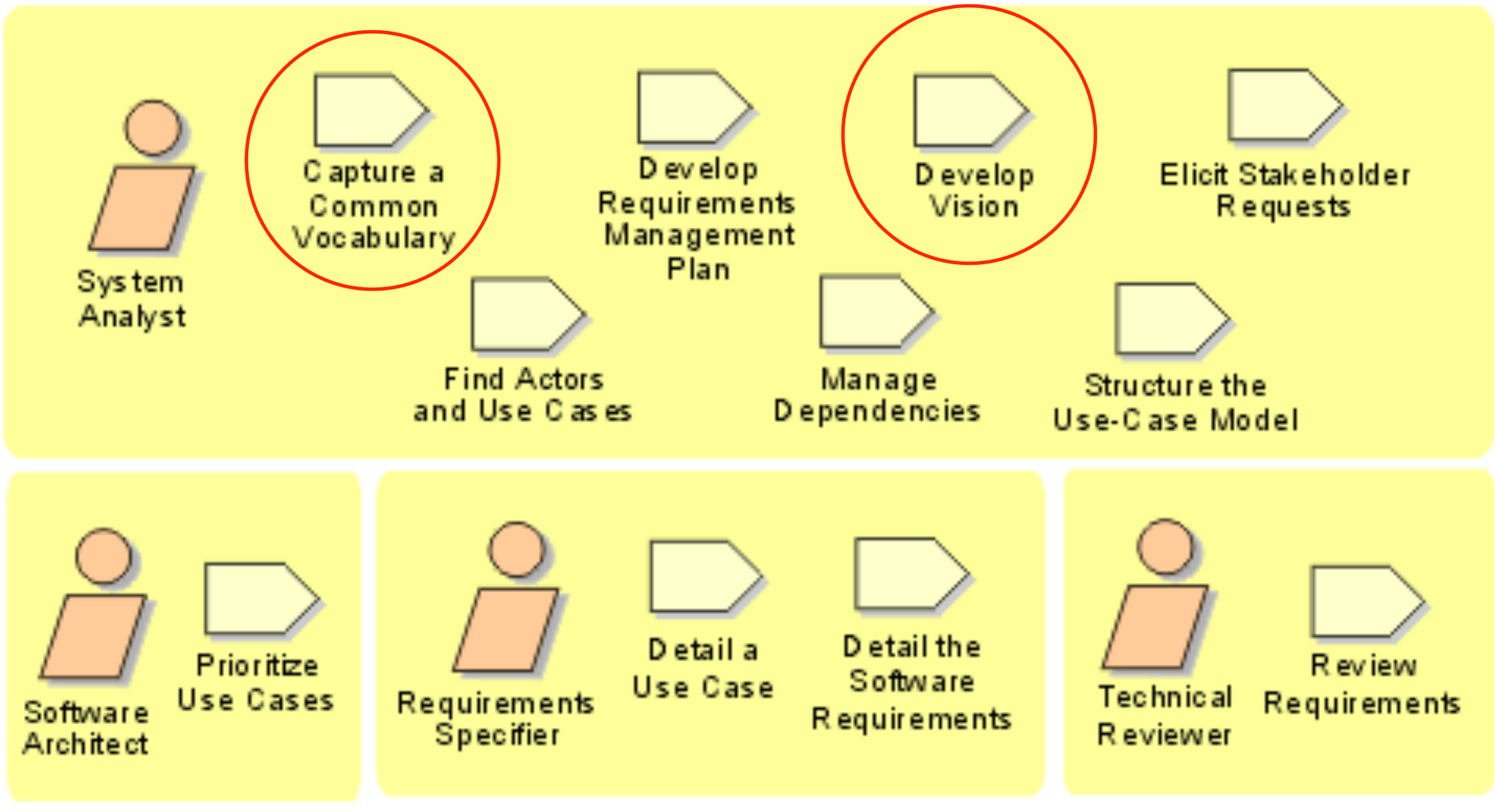
- **User stories + post-it lístečky** – kreslený vzhled GUI a jeho přetváření pomocí nalepovacích štítků (tzv. Post-it).

Proces

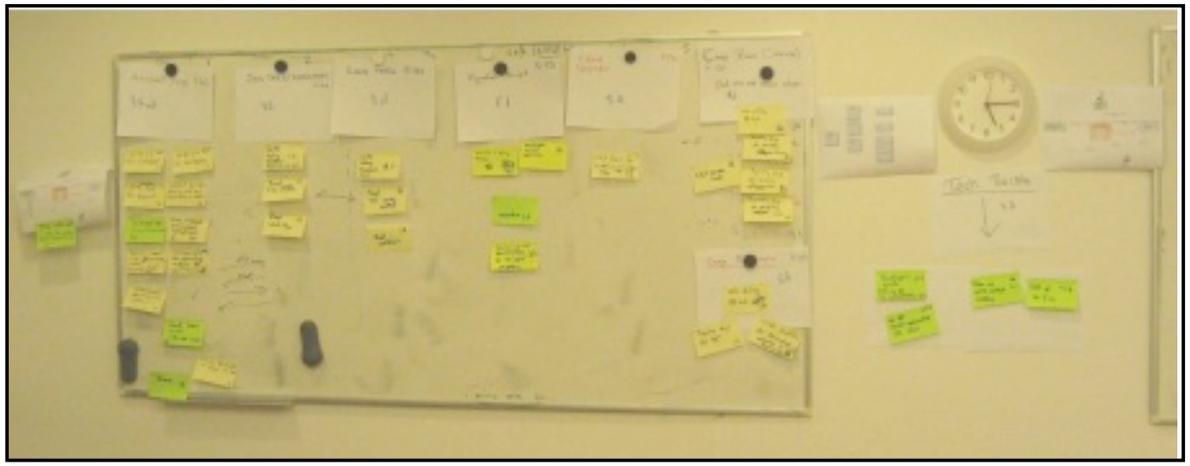

Cost Management
Any change take the correct management plan
Change Request
Write down the change and send to CCB
Change Request Meeting
Evaluate the CR and decides to approve or reject
Approve
Project Management Plan
Update the documents
Project Management Information System
Plan is updated using configuration management system


Artifacts
Business Rule
Vision
SRS
Stakeholder Requests
Use-Case Model
Supplementary Specification
Design Model
Test Suites
End-User Documentation Materials and Training Materials
Traceability Items
Stakeholder/User Needs
Features
Software Requirements
Design / Test / Doc Requirements

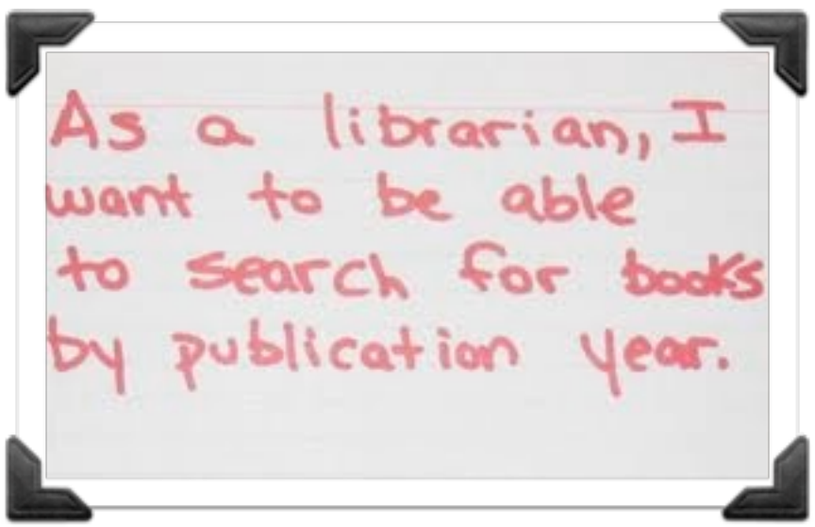
Role a jejich aktivity

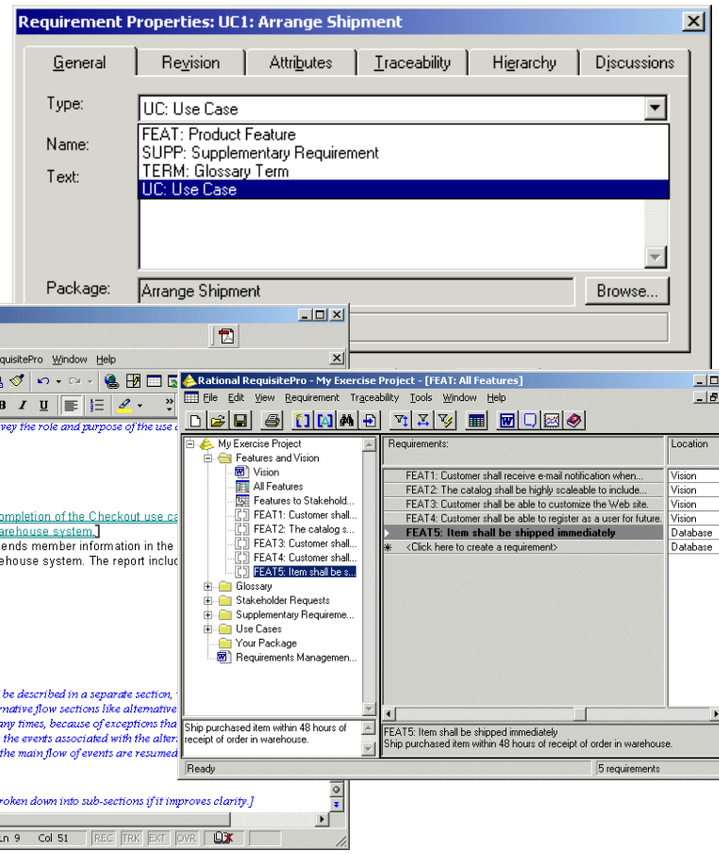


Stupeň formálnosti požadavků



Neformální

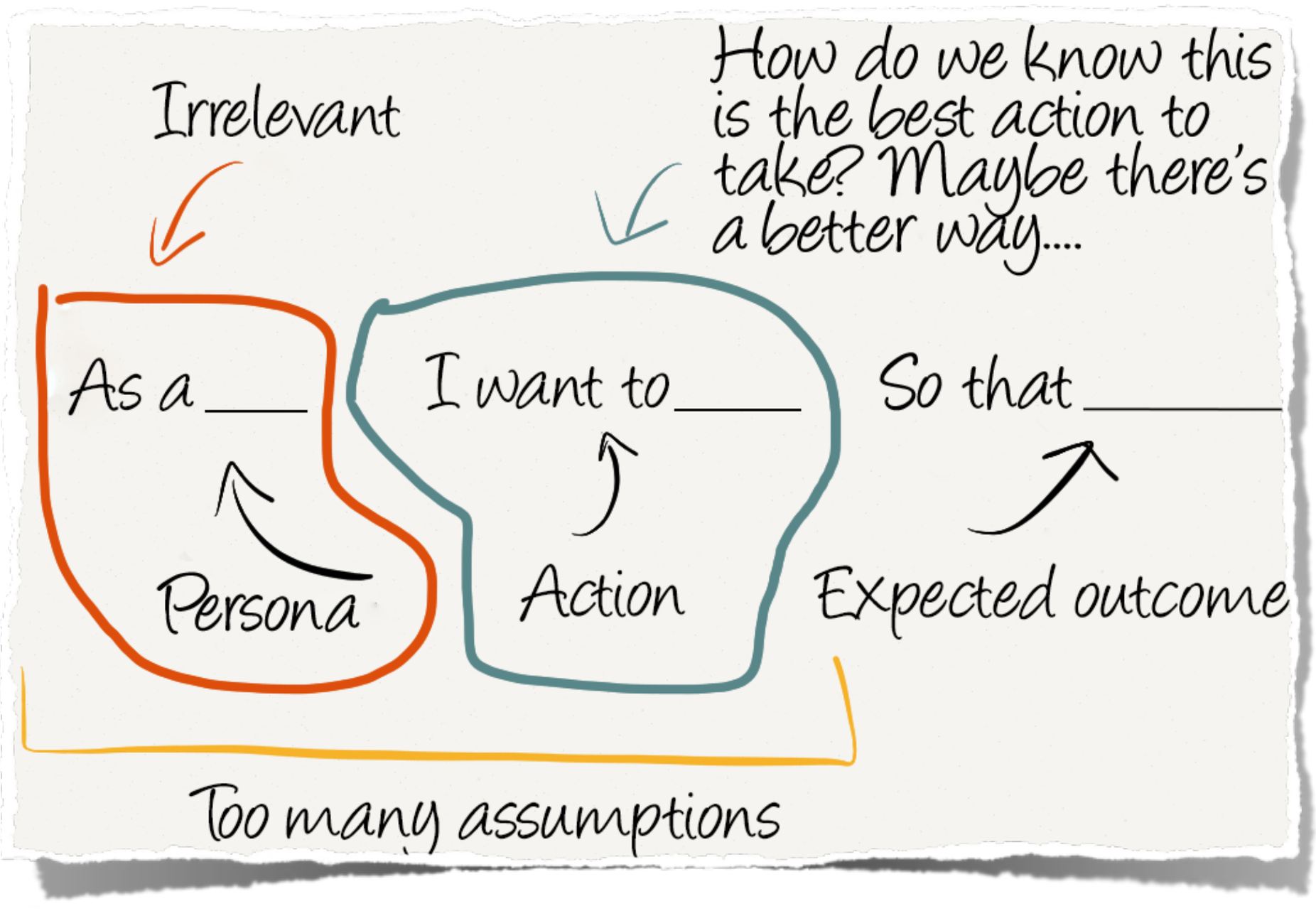




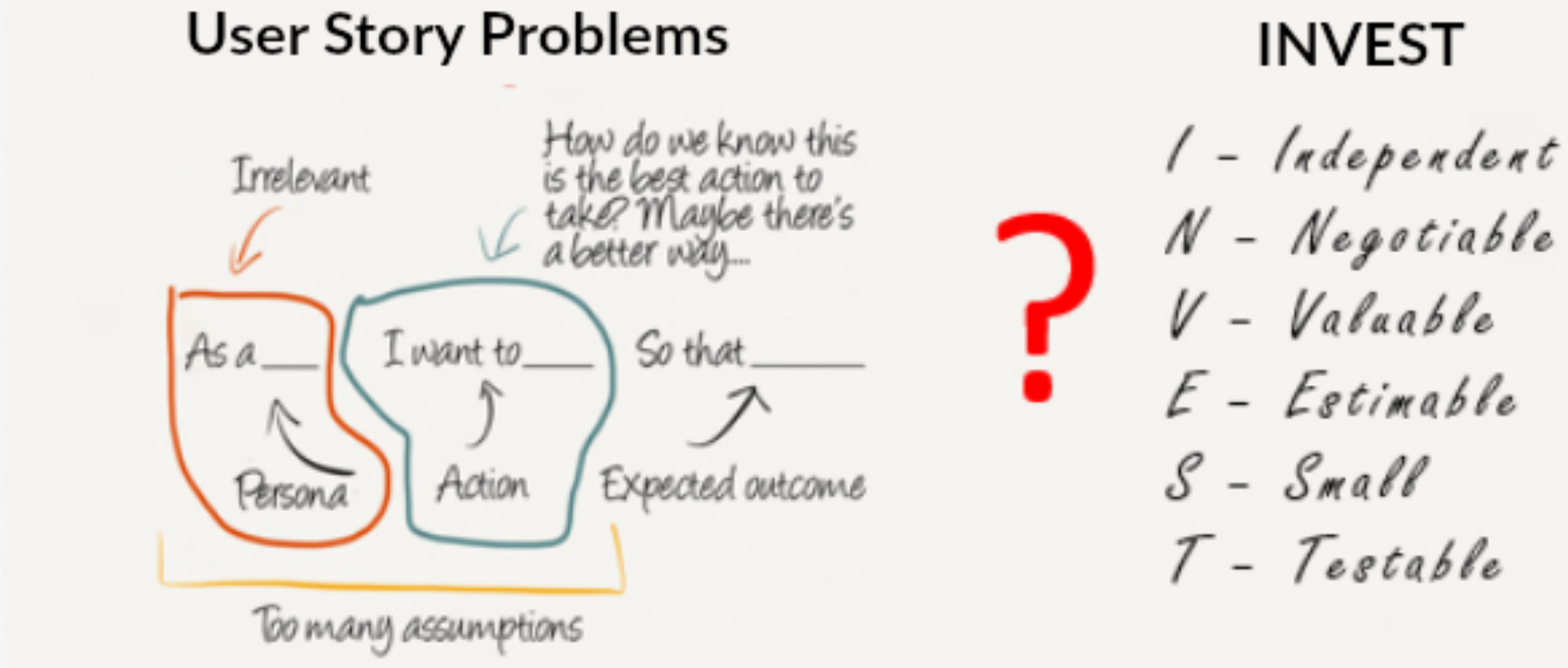
Formální - Rational RequisitePro

User Stories

User Stories

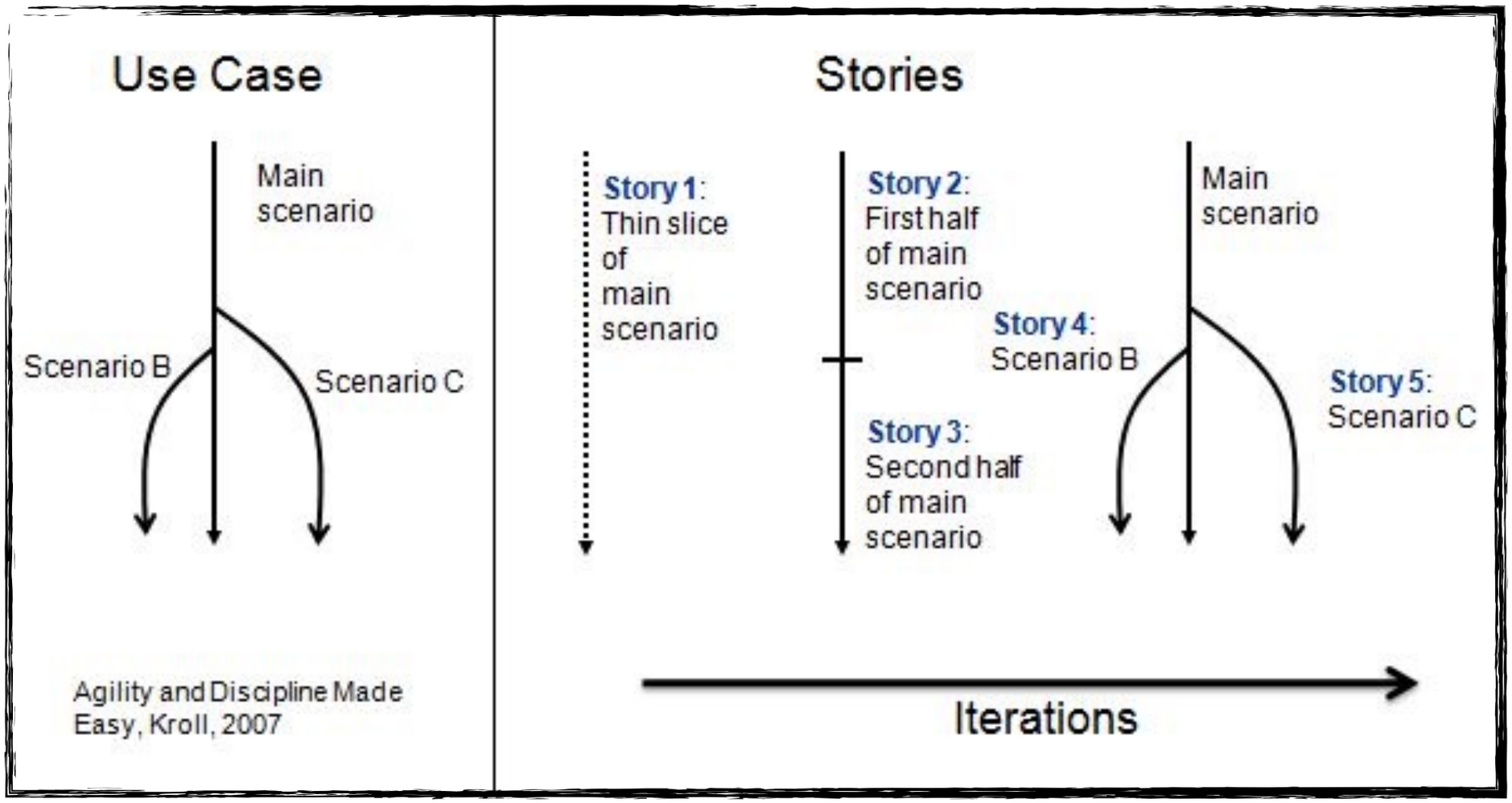


User Stories



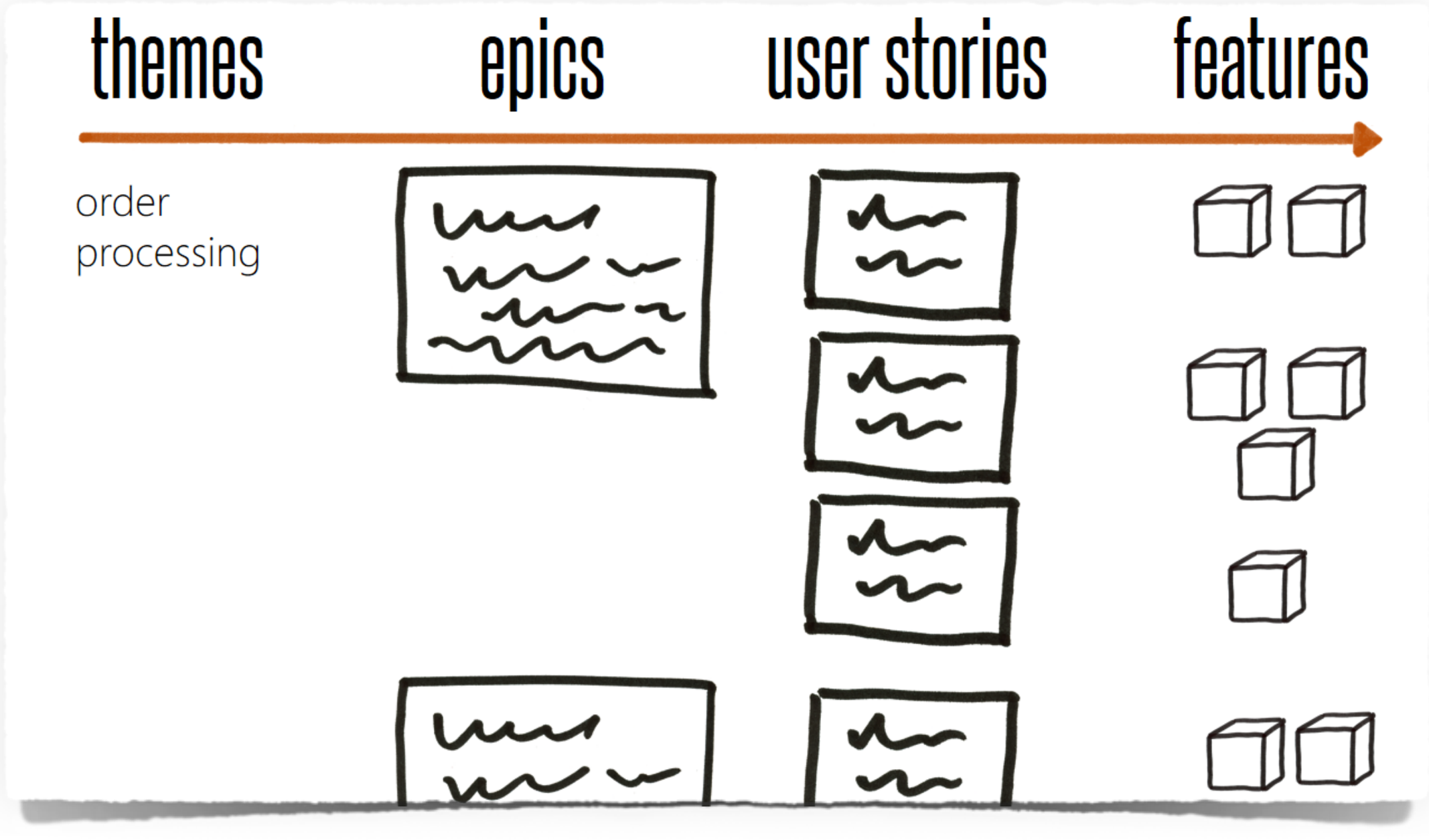
Do you Invest in Your User Stories?

Use Case / User Stories



User Stories

Témata a epiky



Příklady zápisu

- M - Jako manažer chci mít možnost zobrazit si odpracované hodiny pro daný projekt nebo člověka z IS a provést potřebné korekce a doplnění dodatečných informací, abych mohl vygenerovat a poslat fakturu zákazníkovi.

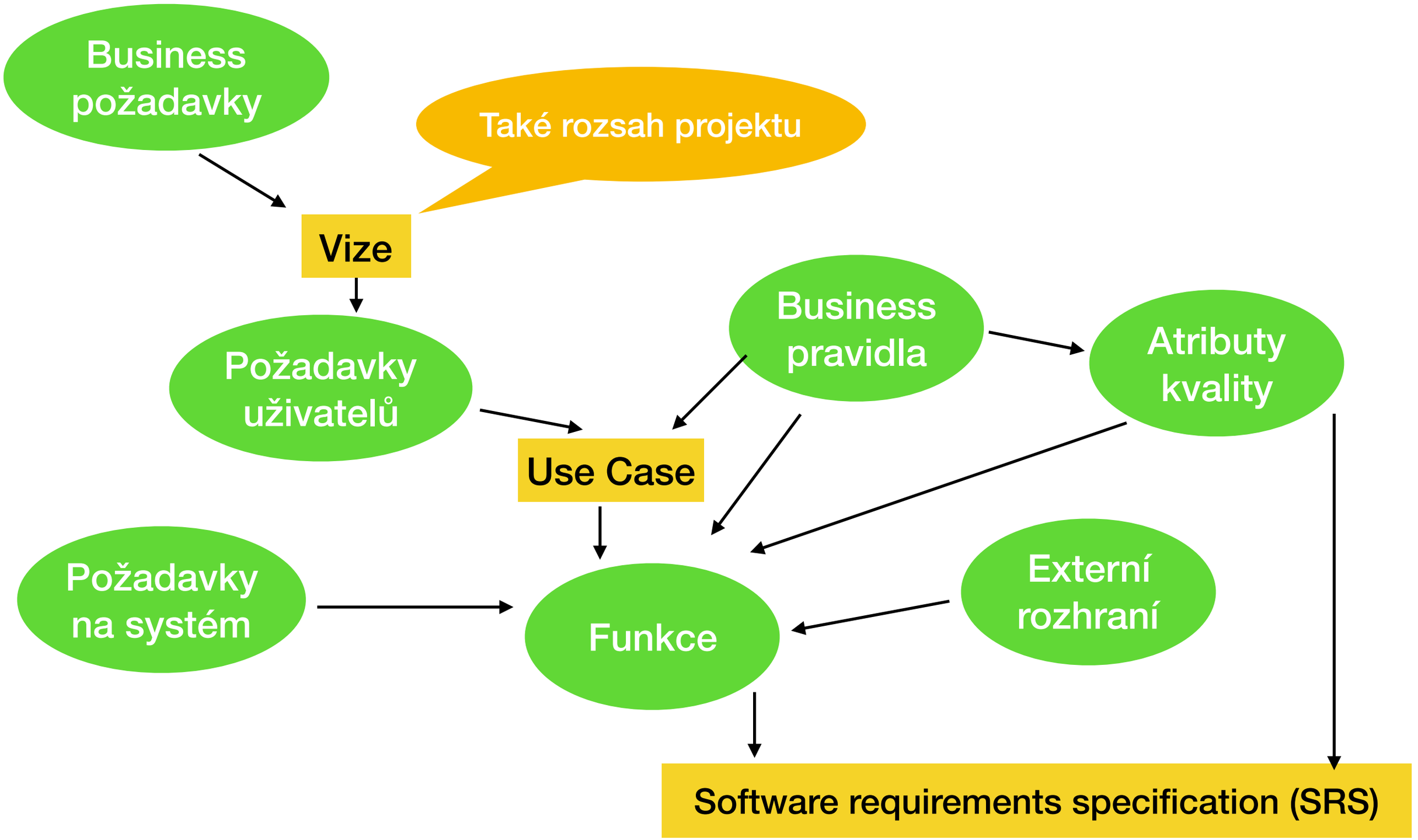
- S - Jako vedení chci mít možnost zobrazit si datové analýzy nad numerickými daty souvisejícími s vyúčtováním (např. odpracované hodiny, vykázané hodiny, vývoj ceny na hodinu, náklady vs příjem, náklady interních projektů nebo za režijní role typu HR atd.).

- C - Jako obchodník chci mít možnost evidovat stav vygenerované faktury, abych mohl uhánět zákazníky

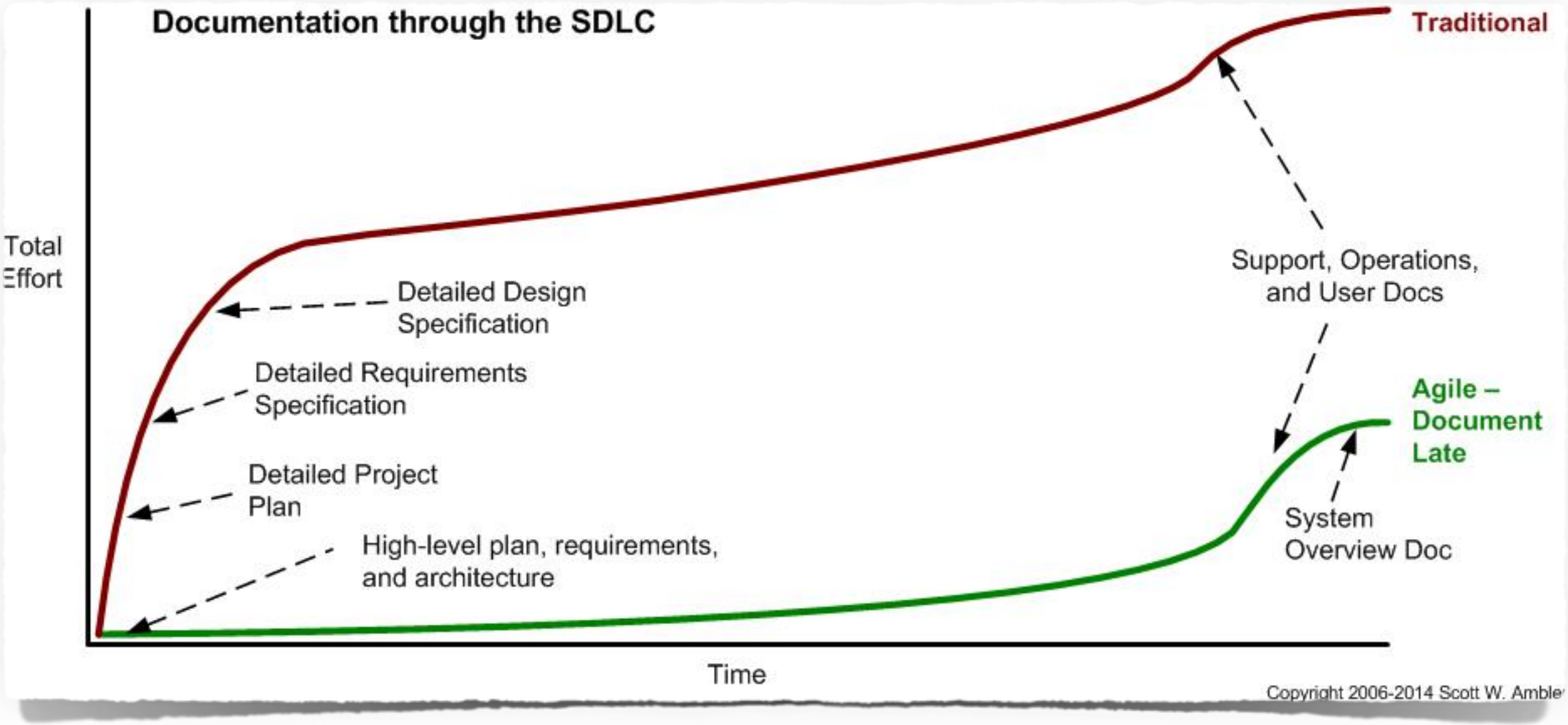
Měření úspěšnosti

- CSF - Critical Success Factor
  - Něco v čem musí být firma úspěšná
  - Např. výborná zákaznická podpora
- KPI - Key Performace Indicator
  - Hodnotí, jakým způsobem naplňujeme CSF
    - počet stížností
    - dotazník zákazníkům
    - počet zopakovaných objednávek

Requirements v kostce



Dokumentace



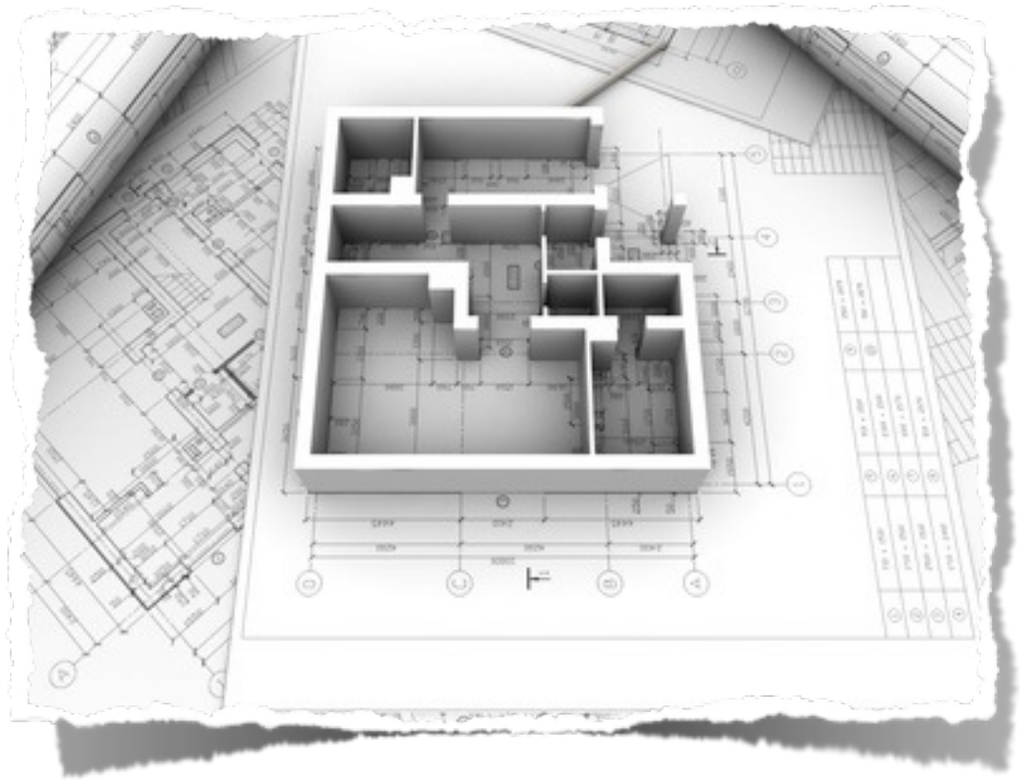
Cvičení

- Dejte týmy dohromady.
- Již máte identifikovány Use Case.
- Také máte identifikovány některé uživatelské scénáře.
- Určete si jeden scénář a v něm najděte jednu funkcionalitu systému.
- Pro tuto funkcionalitu určete atributy kvality - pomozte si šablonou od SEI QAW.
- Výsledek prezentujte.

Cvičení

Scenario Refinement for Scenario N		
Scenario(s):		
Business Goals:		
Relevant Quality Attributes:		
Scenario Components	**Stimulus:**	
	Stimulus Source:	
	Environment:	
	Artifact (If Known):	
	Response:	
	Response Measure:	
Questions:		
Issues:		

Analysis & Design



Návrh nebo Design?

Design = návrh

- Není vytváření použitelného uživatelského prostředí (pouze malinká podmnožina celého návrhu)
- Často takto omezeně chápáno studenty – nedokáží si představit, co všechno návrh obnáší, navrhovali většinou jednoduché aplikace
- Znají návrh vzhledu webových stránek - tvorba GUI je podle nich to hlavní, co musíme dělat i při návrhu SW

  Důsledek: omezení návrhu na tvorbu *GUI* a *databáze*, čímž také často začínají aniž by vůbec uvažovali požadavky na systém.

Analýza a Návrh

Analýza

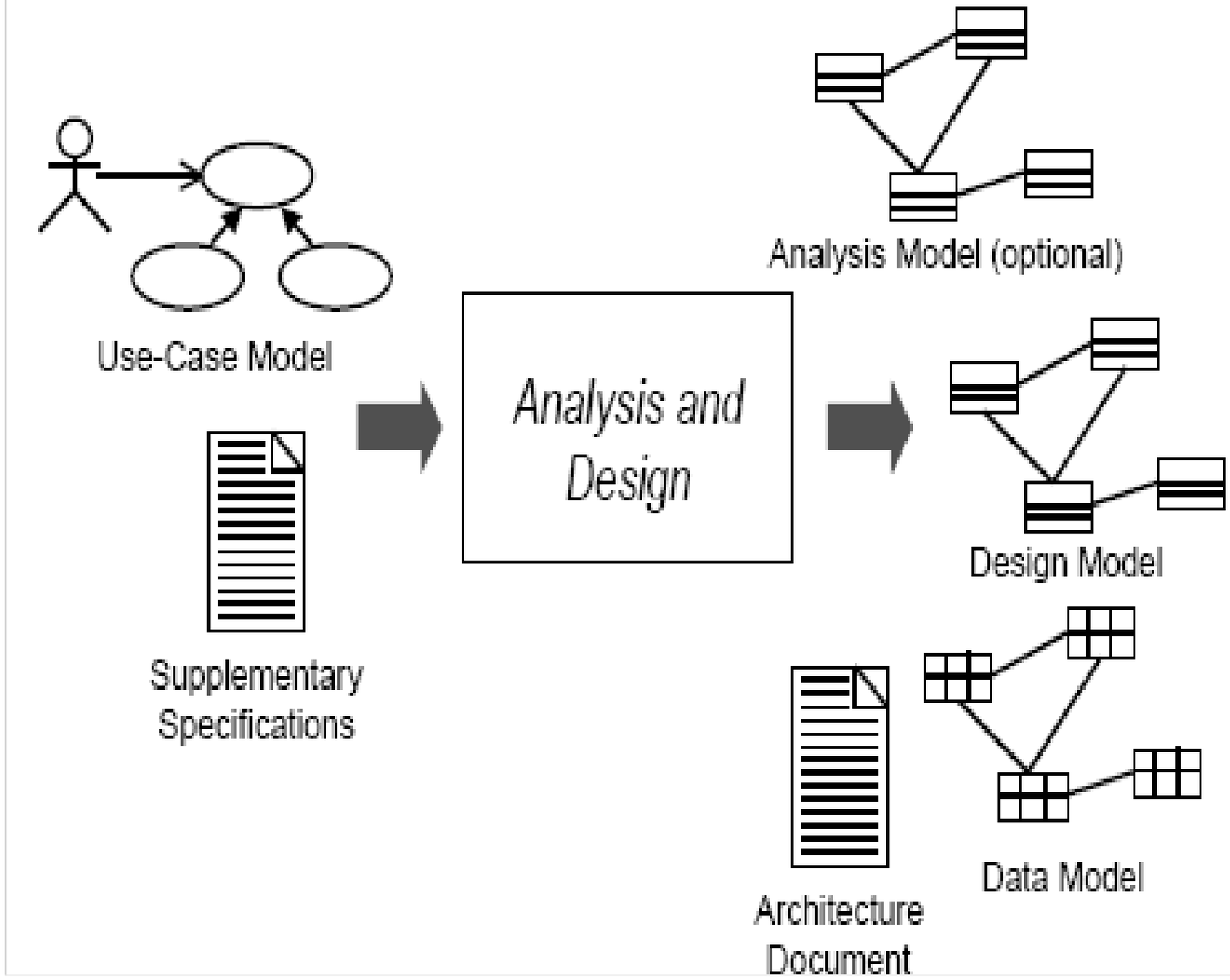
- zaměřuje na pochopení funkčních požadavků na systém,
- zjednodušeně opomíjí nefunkční požadavky, implementační omezení a prostředí.
- výsledkem analýzy je téměř ideální obraz budoucího systému ve formě množiny tříd a subsystémů, ale bez vazby na technologii (neuvažujeme knihovny, kolekce, databáze)

Návrh

- výsledky analýzy mapovány na implementační prostředí a na omezení plynoucí z nefunkčních požadavků.
- další krok k výsledné aplikaci, další zjemnění, propracování analýzy.
- cílem návrhu je mít optimálně navržený systém pokrývající veškeré požadavky, a to jak funkční, tak nefunkční.

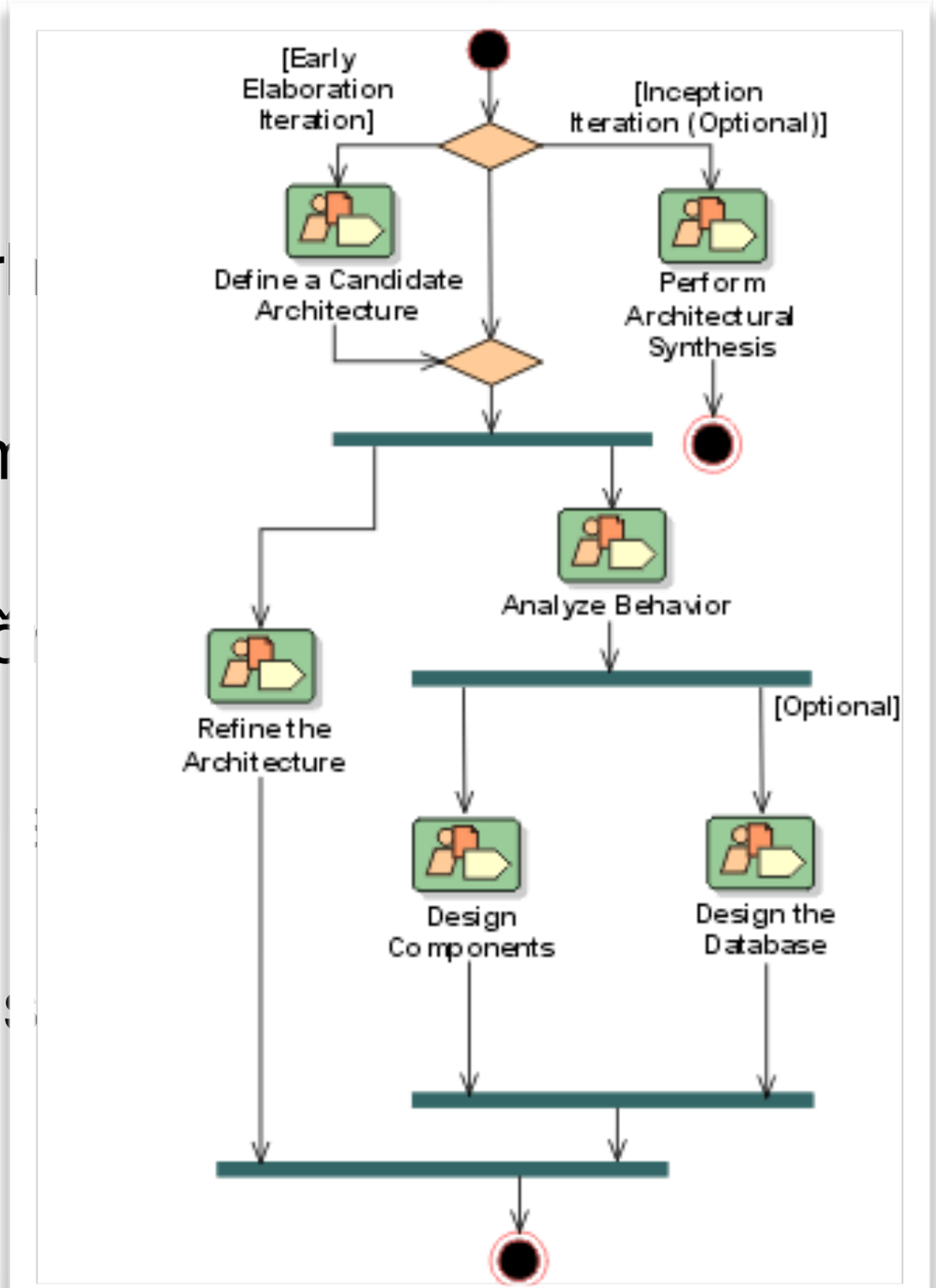
Analýza a návrh řízený Use Case

Kde jsou požadavky uživatele?



Analysis & Design

- Transformace požadavků do návr
- Vývoj robustní architektury systém
- Přizpůsobení návrhu implementač
- Architektura (komunikace, vrstvy,
- Modely – druhy, formálnost (user



Průstup workflow životním cyklem vývoje

- Inception – architekt se spolupodílí na definici a prioritách UC (kritické, nejrizikovější nejdříve)
- Elaboration – návrh těchto UC (a následná implementace – jiná disciplína), odstraňování rizik, prototypy
- Architekt – definice a kontrola dodržování mechanismů (komunikace, ukládání, vrstvení)
    - Guidelines
    - Workshopy
    - Review
- Spolupráce architektů s analytiky a vývojáři

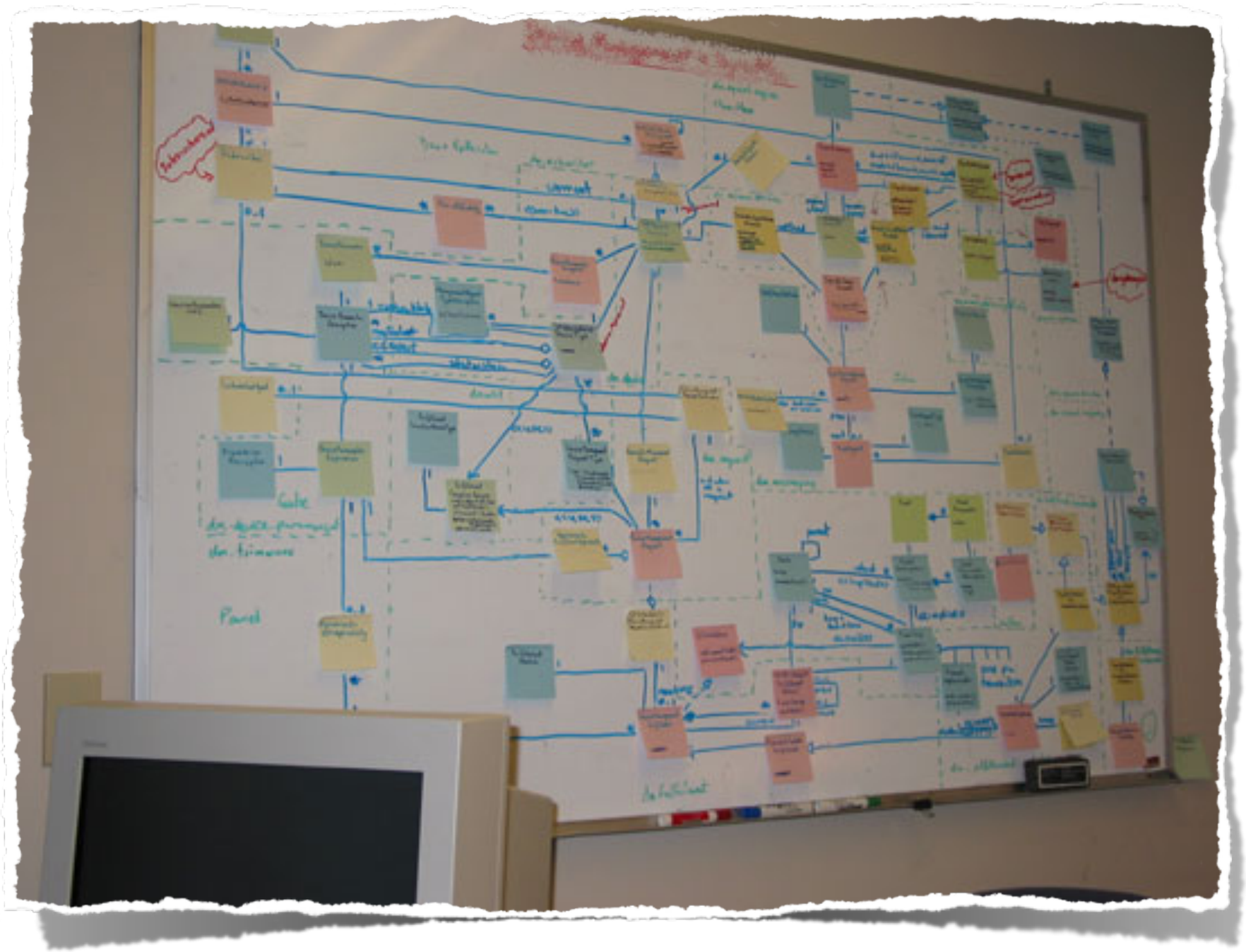
Architektura aplikace

- Co je to architektura?
- Důsledky její neexistence/špatného návrhu?

Nejčastější architektury

- Vrstvená
- Roury a filtry
- Publisher-Subscriber
- Implicit-Invocation (IoC)
- Síťově-centralizovaná architektura
  - P2P
  - Client-Server

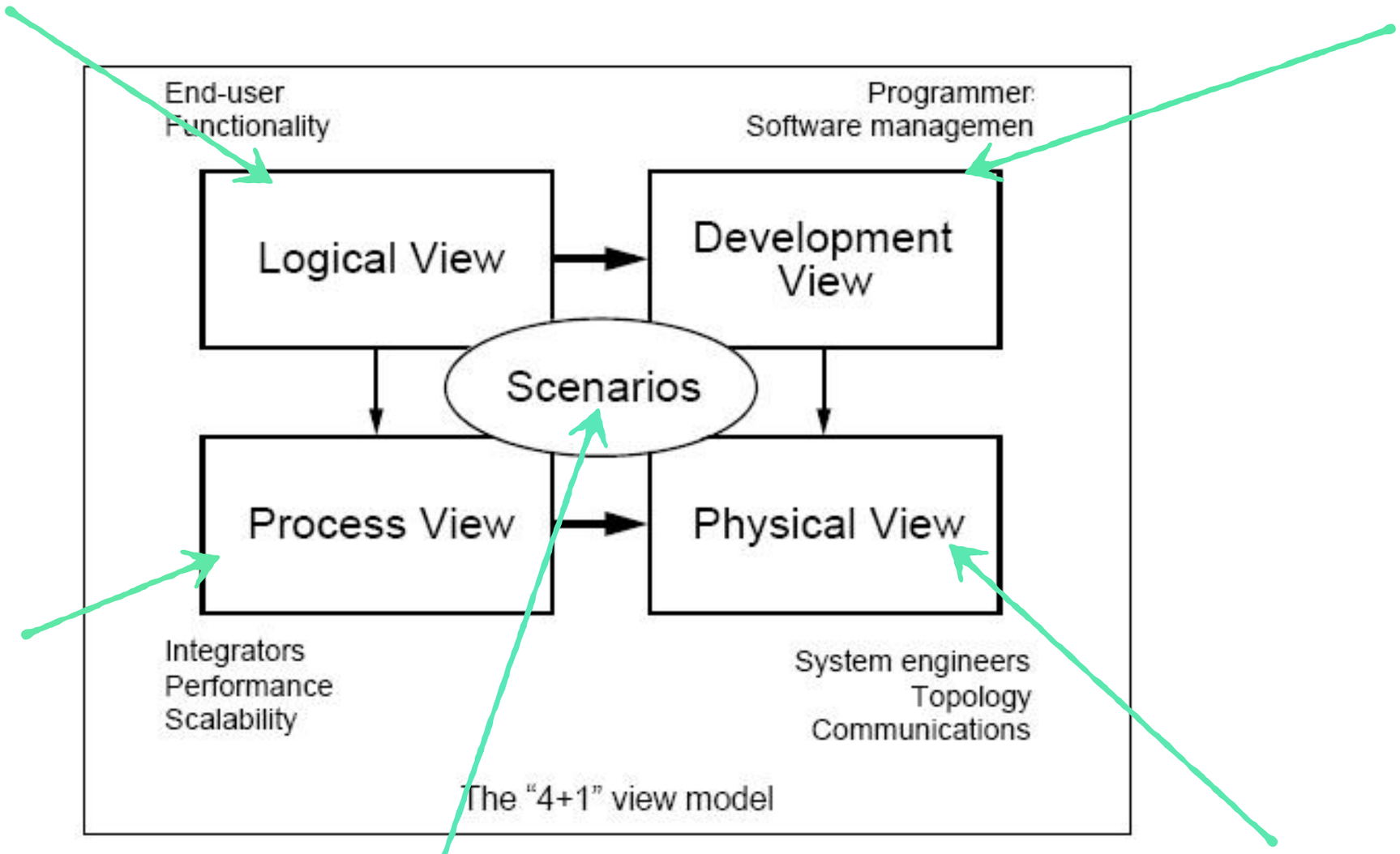
Architektura - modely



Architektura 4+1 (Kruchten)

2. Konceptuální chování

3. Balíčky, subsystémy



+1 Procesy, thready, systémy

1. Use Case model

4. Model nasazení

SEI architektura

Module **Component and Connector** **Allocation**

Client-Server **Shared Data**

Class **Decomposition**

Process **Deployment**

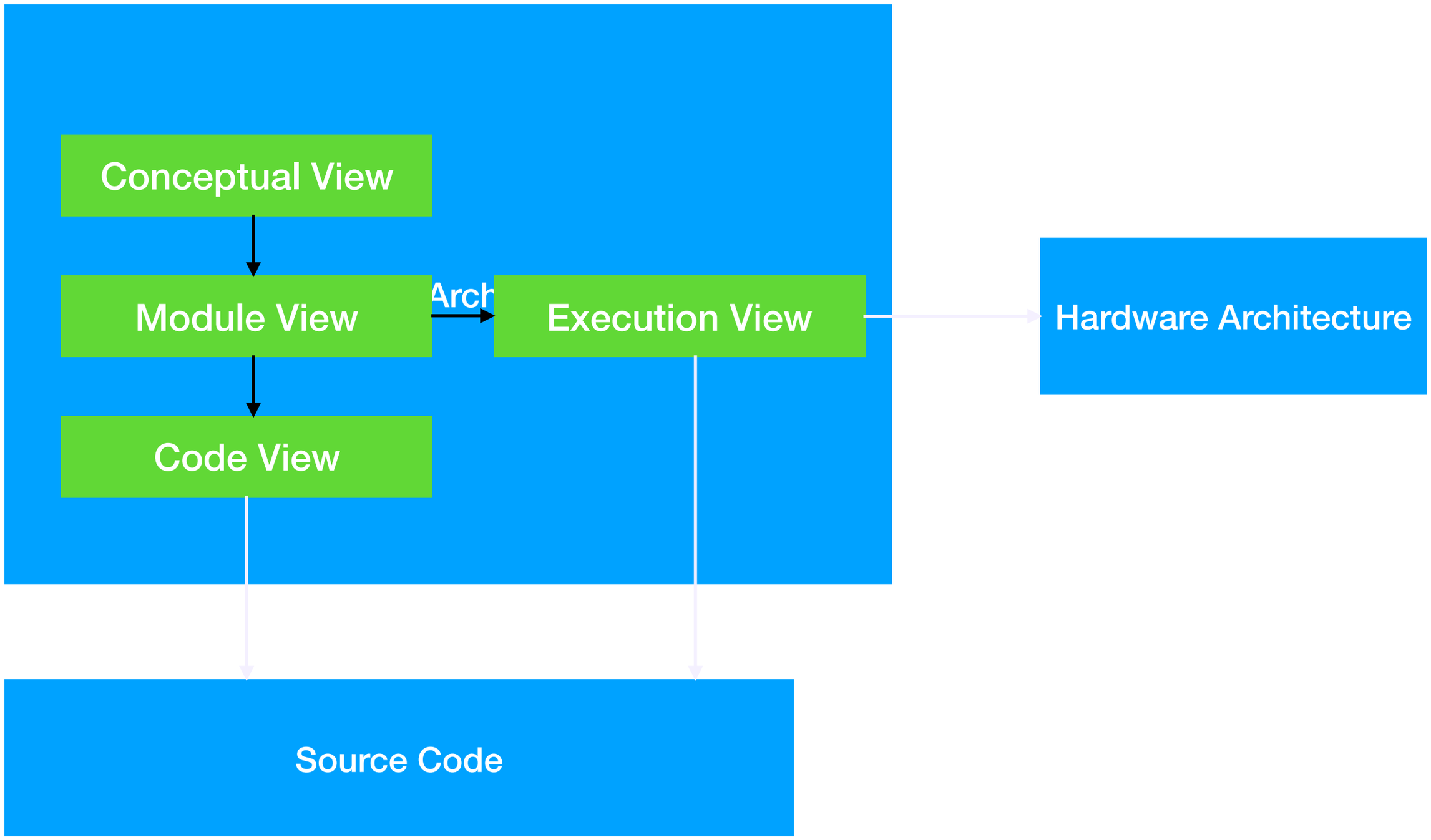
Uses

Concurrency

Implementation

Layered **Work Assignment**

Siemens

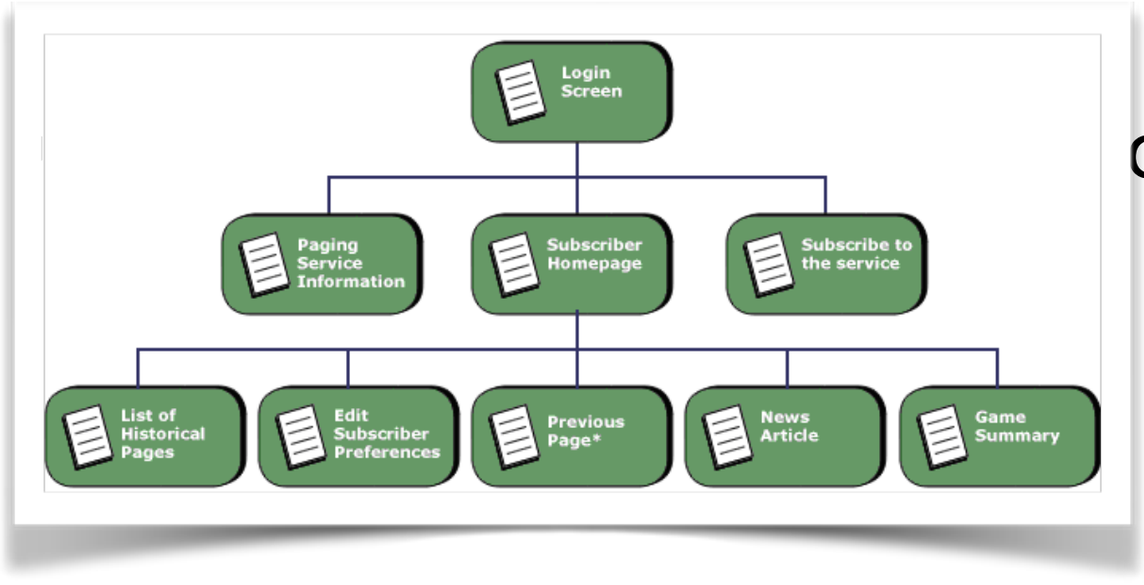

Conceptual View
Module View
Arch
Execution View
Hardware Architecture
Code View
Source Code

Testovatelnost a architektura

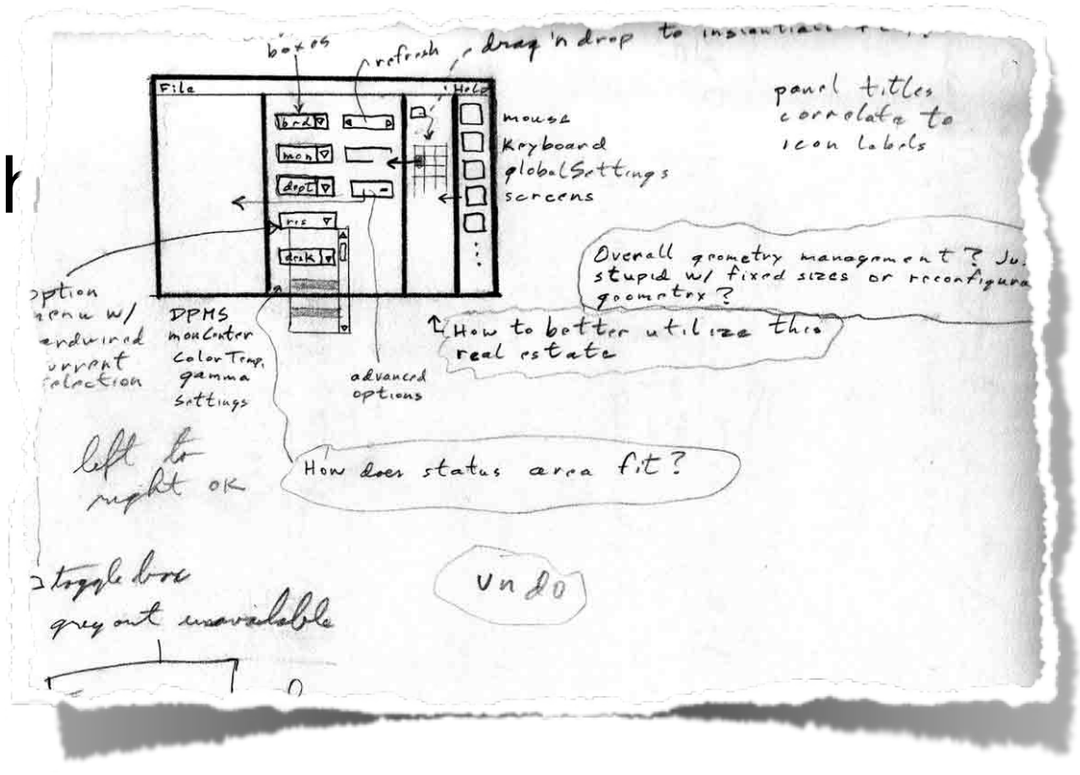
- Úzce provázáno, vliv architektonických částí na testování:
    - Použitá technologie (srovnejte možnosti: Java vs. Pascal)
    - SW architektura - logické vrstvy (monolitická, 2, 3 vrstvá)
    - Způsob komunikace (RMI, TCP, SOAP, …)
    - UI (příkazová řádka, web based GUI, formulářové GUI)
- Automatizace testování, dostupnost nástrojů
    - xUnit
    - Ant
    - Maven
    - Gulp, Phantomjs, Karma

UI - User Stories, mapy

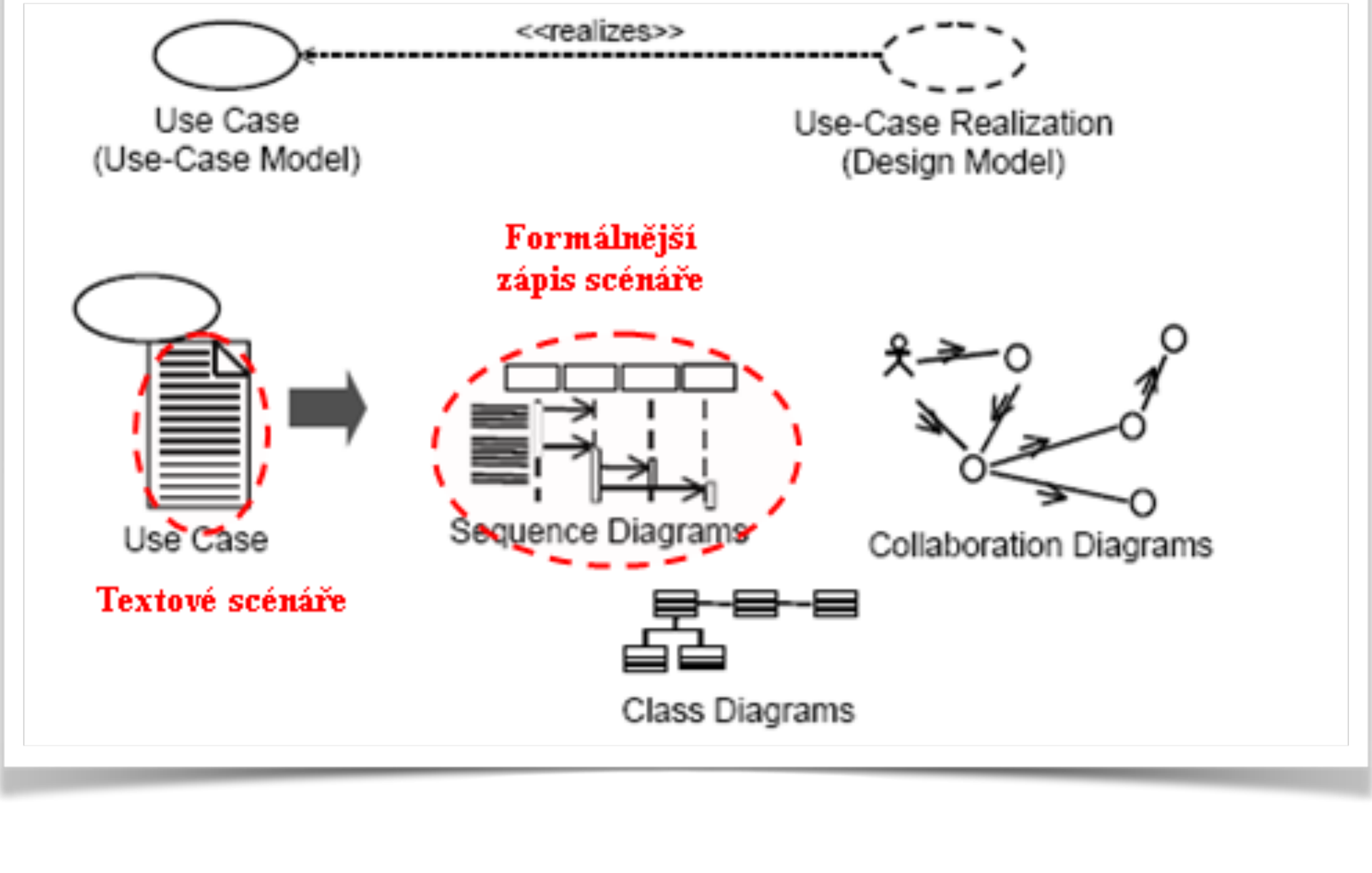
- **User Story** - budoucí aplikace a její stručně popsané použití



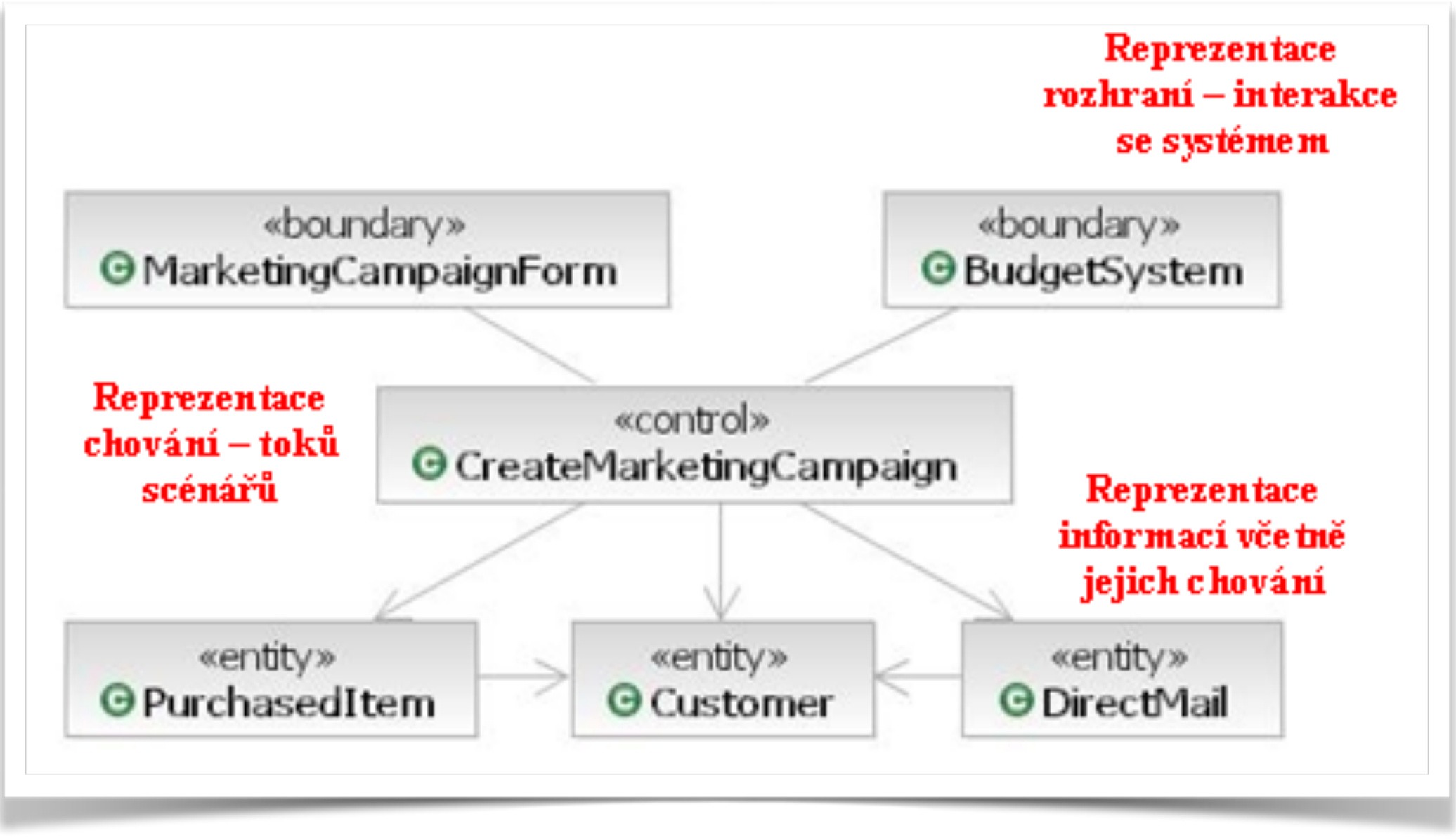
ozł



UC Realizace 1



UC Realizace 2



Artefakty disciplíny

- **Analytický model** (analysis model).
- **Návrhový model** (design model).
- Software architecture document
  - architektura systému
  - několika rozdílných pohledů (view) na systém
  - cílem je zachytit různé aspekty systému
    - ✓ statický pohled (vrstvy, komponenty, subsystémy),
    - ✓ dynamický pohled (jak spolu tyto komunikují),
    - ✓ pohled rozmístění komponent/částí systému na HW komponenty.

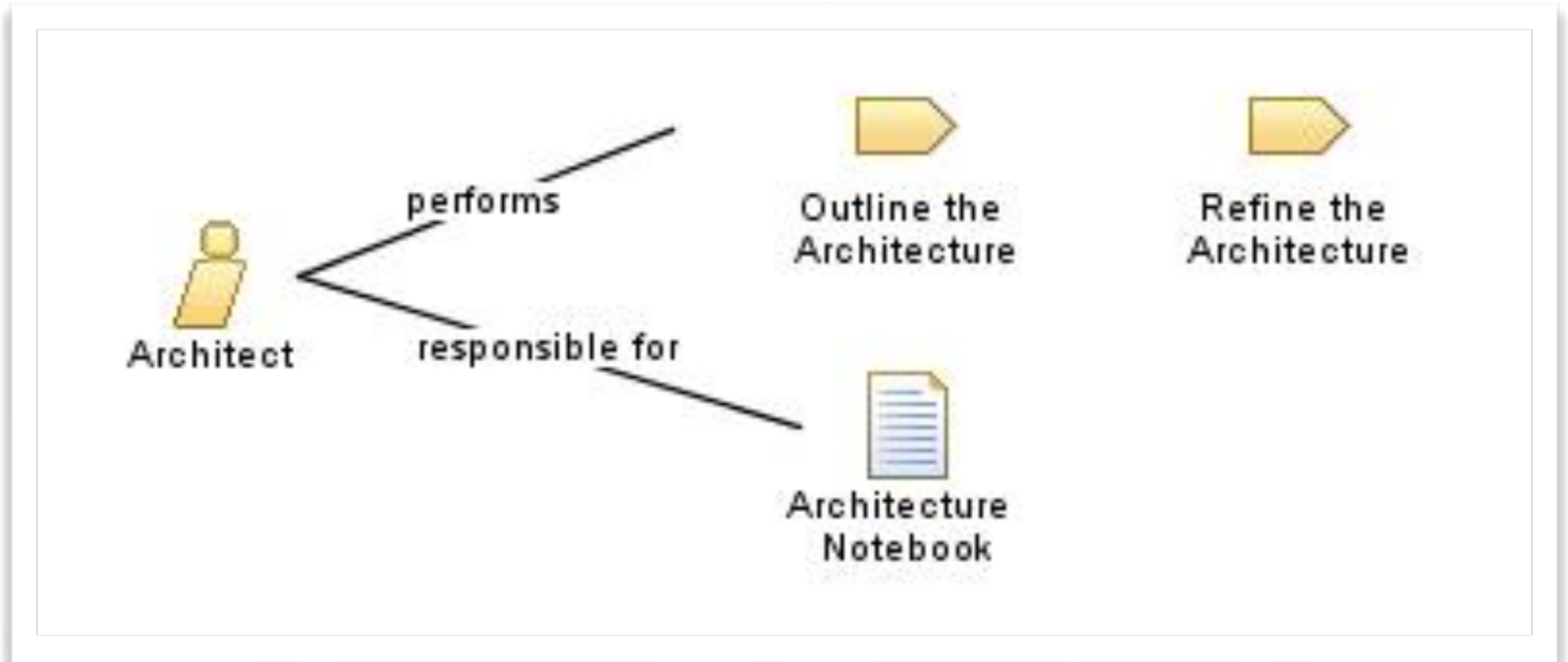
Artefakty disciplíny

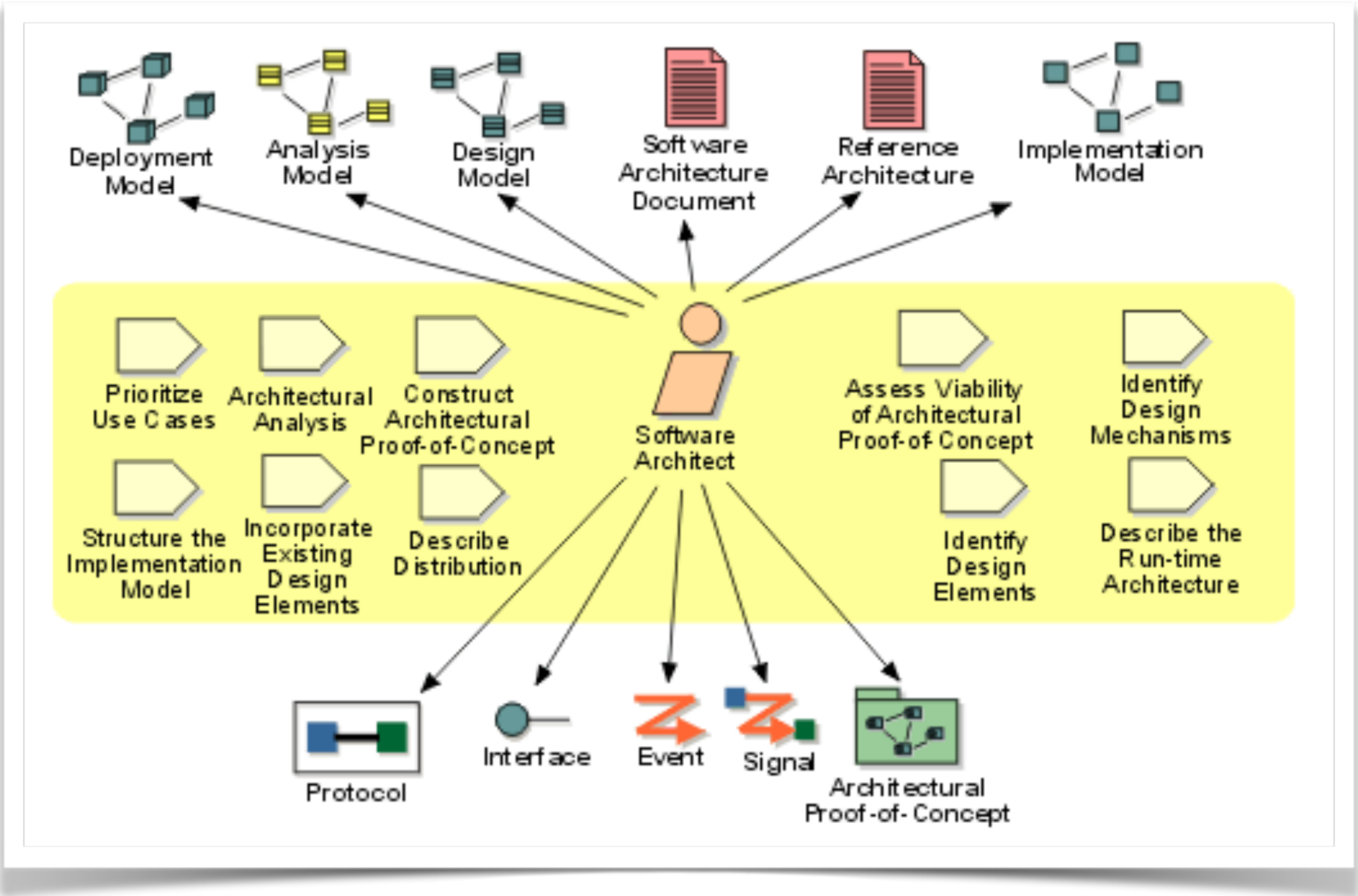
- **Rozhraní** (interface) – specifikace rozhraní pro jednotlivé vrstvy a komponenty systému (sada operací bez detailů jejich implementace).

- **Model nasazení** (deployment model).

- **Datový model** (data model) – je-li nezbytně nutný, vidíte, není důležitý, natož abychom s ním začínali implementaci!

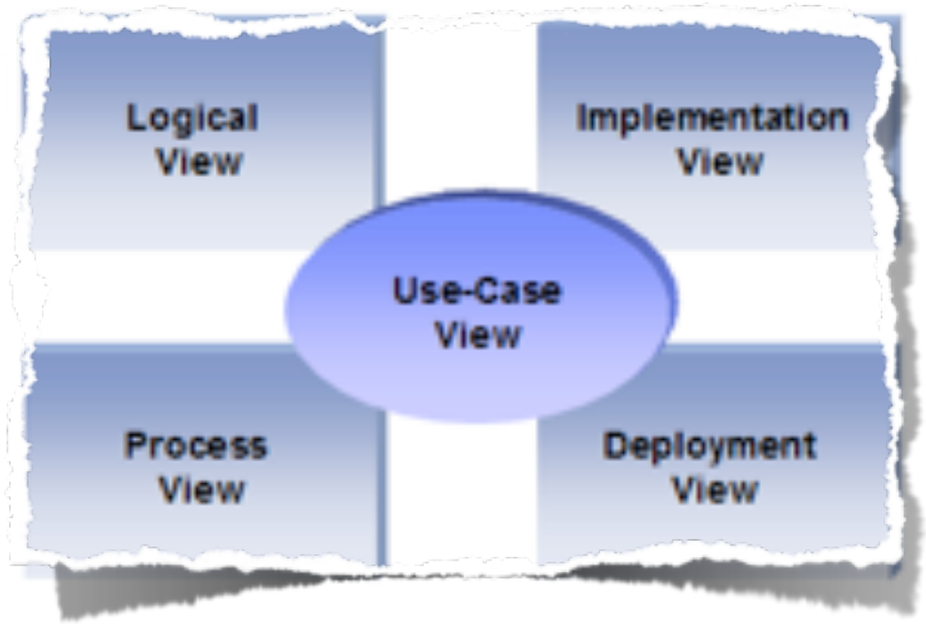
Architekt podle Open-up




Deployment Model
Analysis Model
Design Model
Software Architecture Document
Reference Architecture
Implementation Model
Prioritize Use Cases
Architectural Analysis
Construct Architectural Proof-of-Concept
Software Architect
Assess Viability of Architectural Proof-of-Concept
Identify Design Mechanisms
Structure the Implementation Model
Incorporate Existing Design Elements
Describe Distribution
Identify Design Elements
Describe the Run-time Architecture
Protocol
Interface
Event
Signal
Architectural Proof-of-Concept

Software Architecture Document (Architecture notebook)

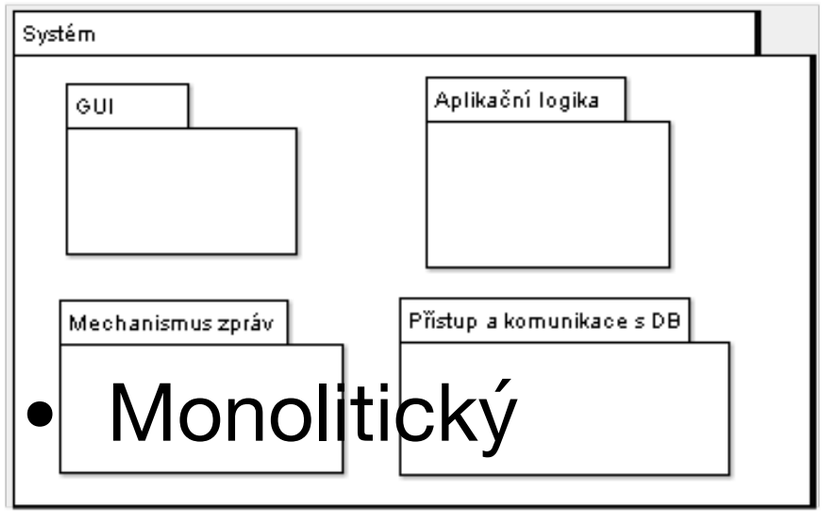
- Úvod (popis, motivace, rozsah)
- Definice jednotlivých bodů (zpravidla 4+1)
- Use Case
- Logický pohled
- Procesní pohled
- Deployment
- Implementace
- Datový pohled
- Výkonnost
- Kvalita



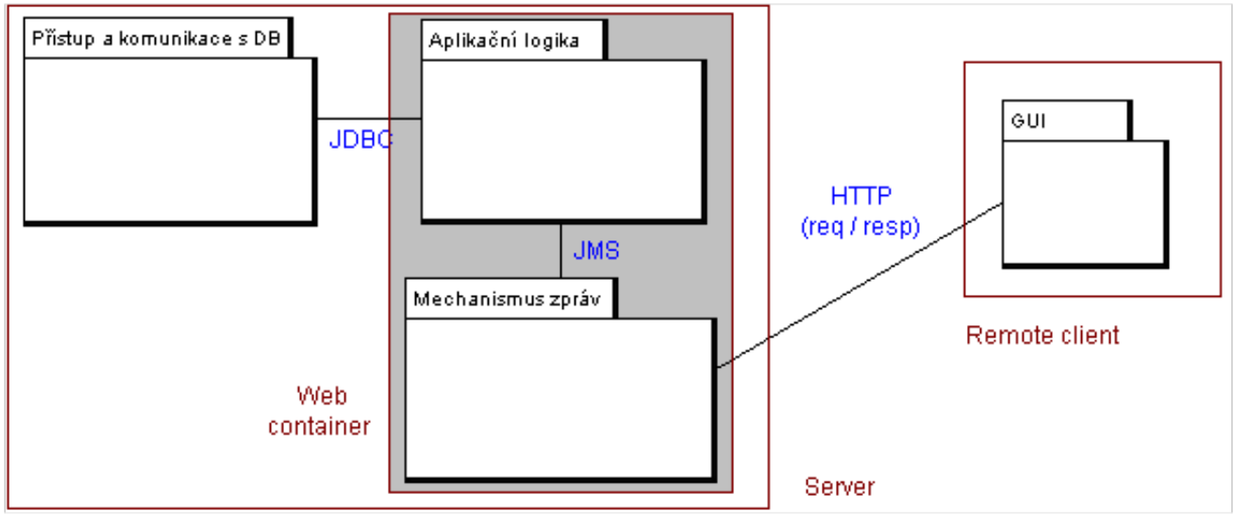
Software architecture document ->

Návrh a komponenty

- Monolitický



Vrstvený



Ortogonalita systému (nezávislé, co nejméně provázané části systému s přesně definovanými odpovědnostmi = rozhraními) přispívá k lepší údržbě a rozšiřitelnosti kódu, zásadní vliv na výkon systému.

Ortogonalita: požadavek na změnu = potřeba provést ji pouze na jednom místě, např. v jedné komponentě (klasicky časté změny v GUI beze změny aplikační logiky).

Cvičení

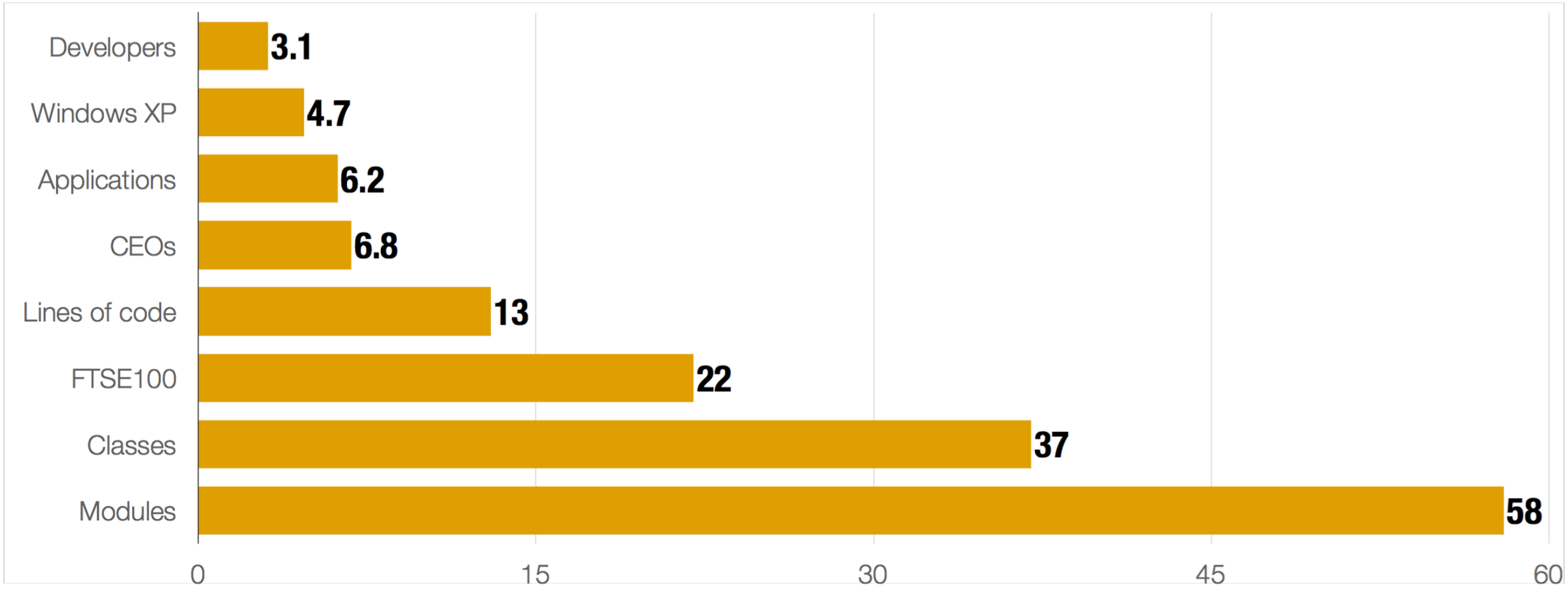
- Dejte týmy dohromady.
- Vzpomeňte si na váš projekt v ROPR. Berme v úvahu koncept 4+1.
- V týmu si rozdělte jednotlivé pohledy - každý zpracuje jeden pohled z architektury 4+1 (fyzický, programátorský, procesní…).
- Výsledek prezentujte v podobě diagramu.

Implementation

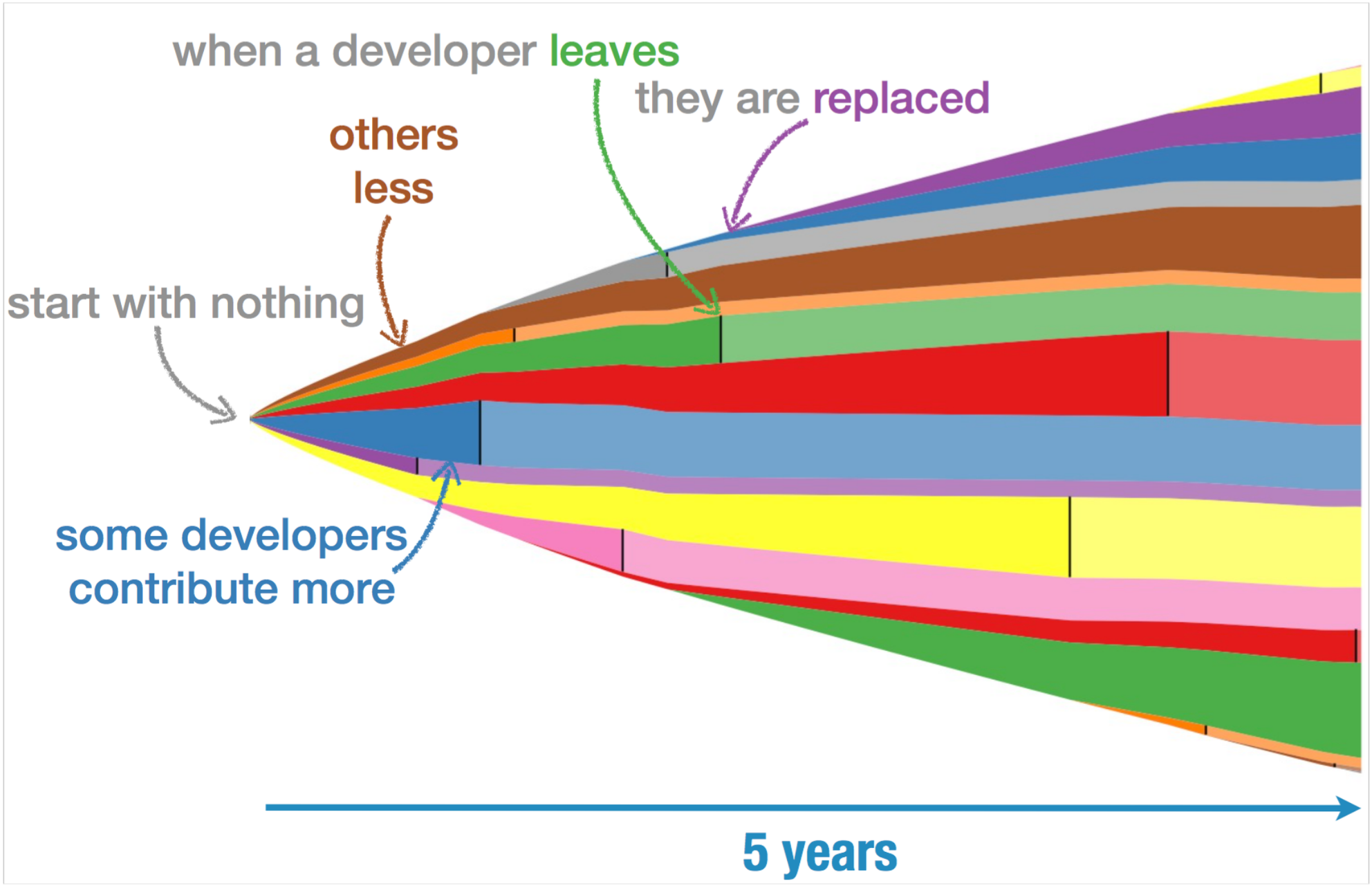
Cíle implementace

- Nejedná se pouze o psaní kódu.
- Nedílnou součástí disciplíny je také:
  - unitové testování
  - integrace komponent
  - nasazení

Životní cyklus firmy



Sedmičlenný tým po dobu pěti let



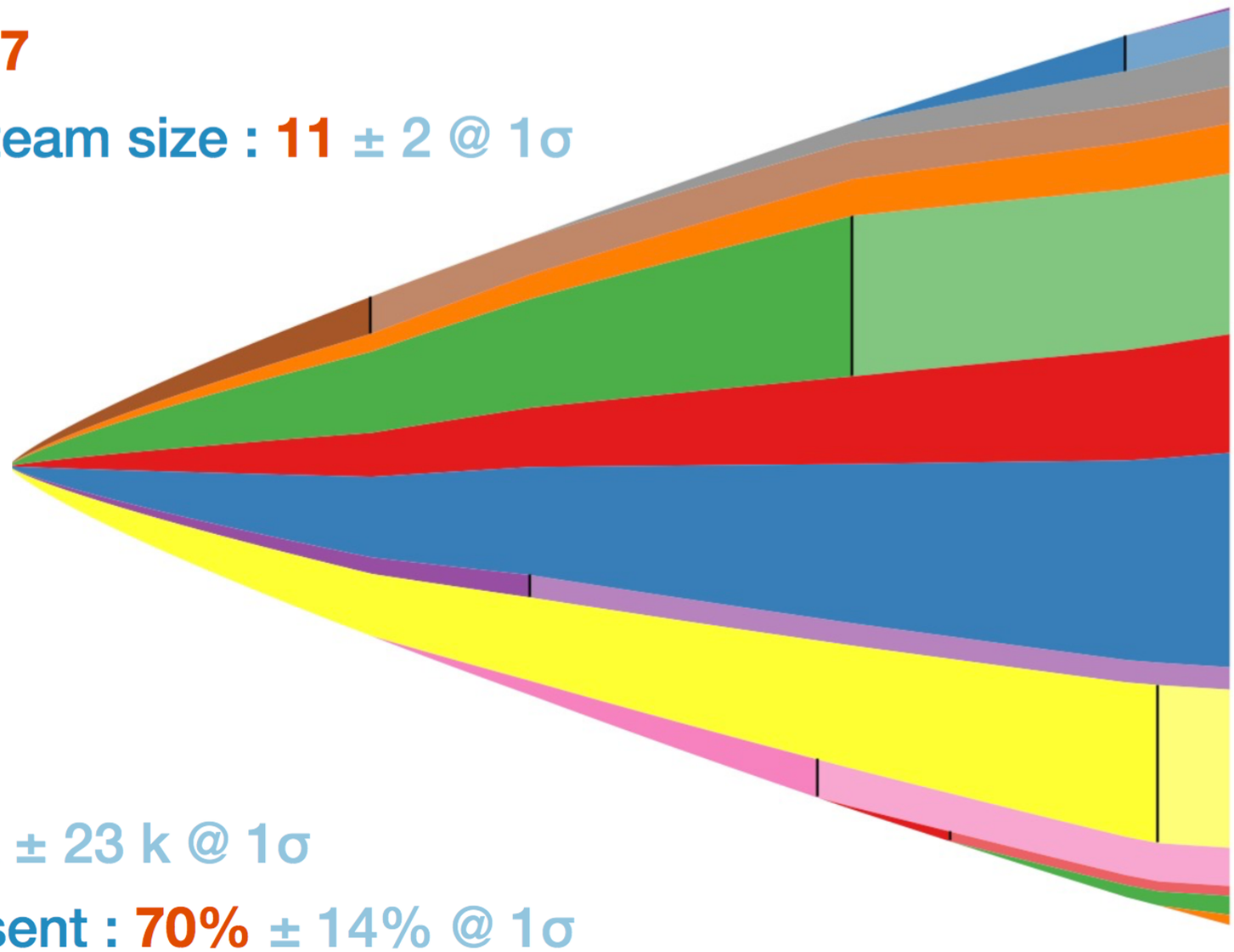
235 000 řádků kódu napsáno 19 lidmi

Pouze 37% kódu napsal současný tým

3 years

Team Size : 7

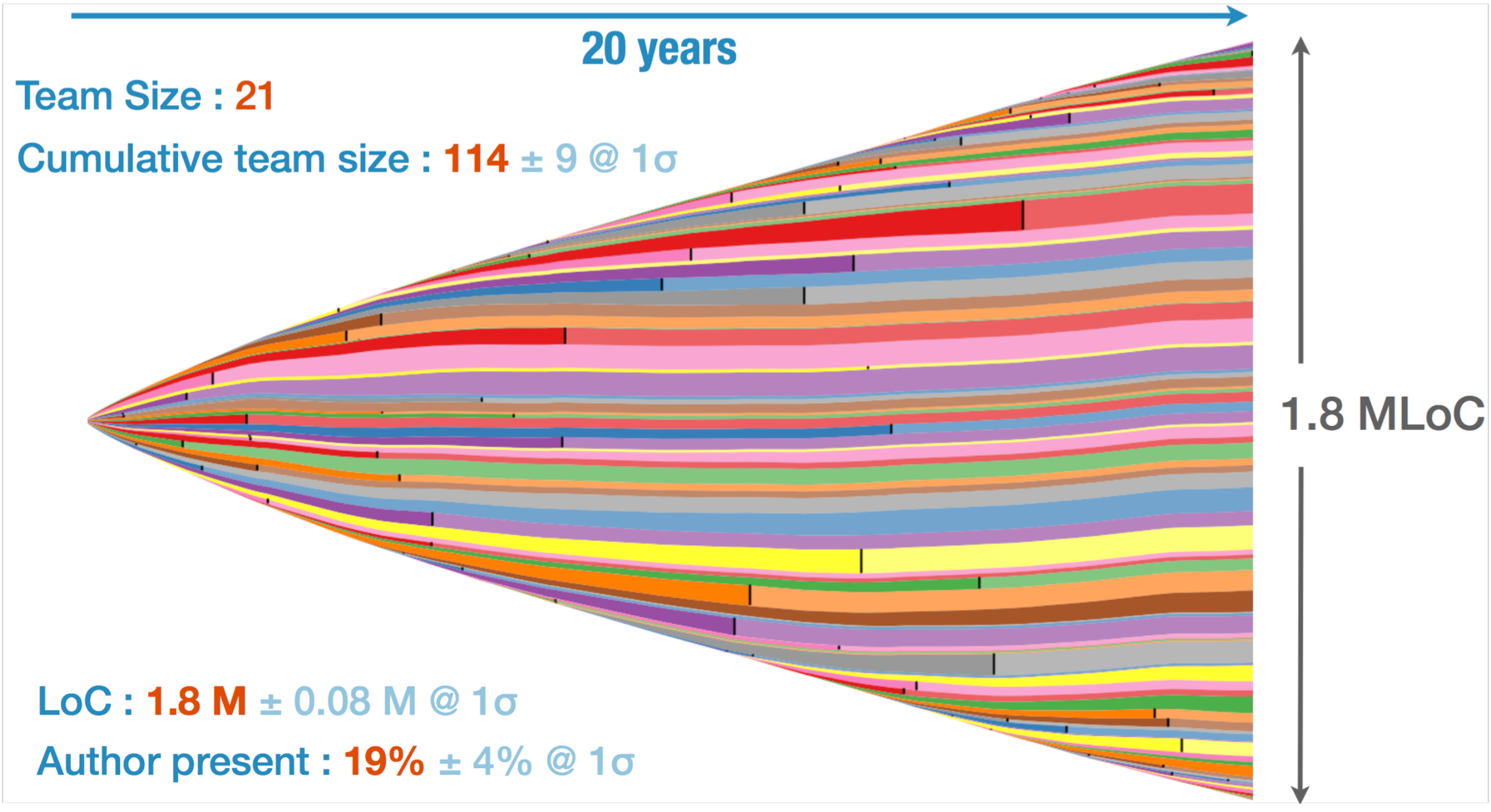
Cumulative team size : 11 ± 2 @ 1σ



157 kLoC

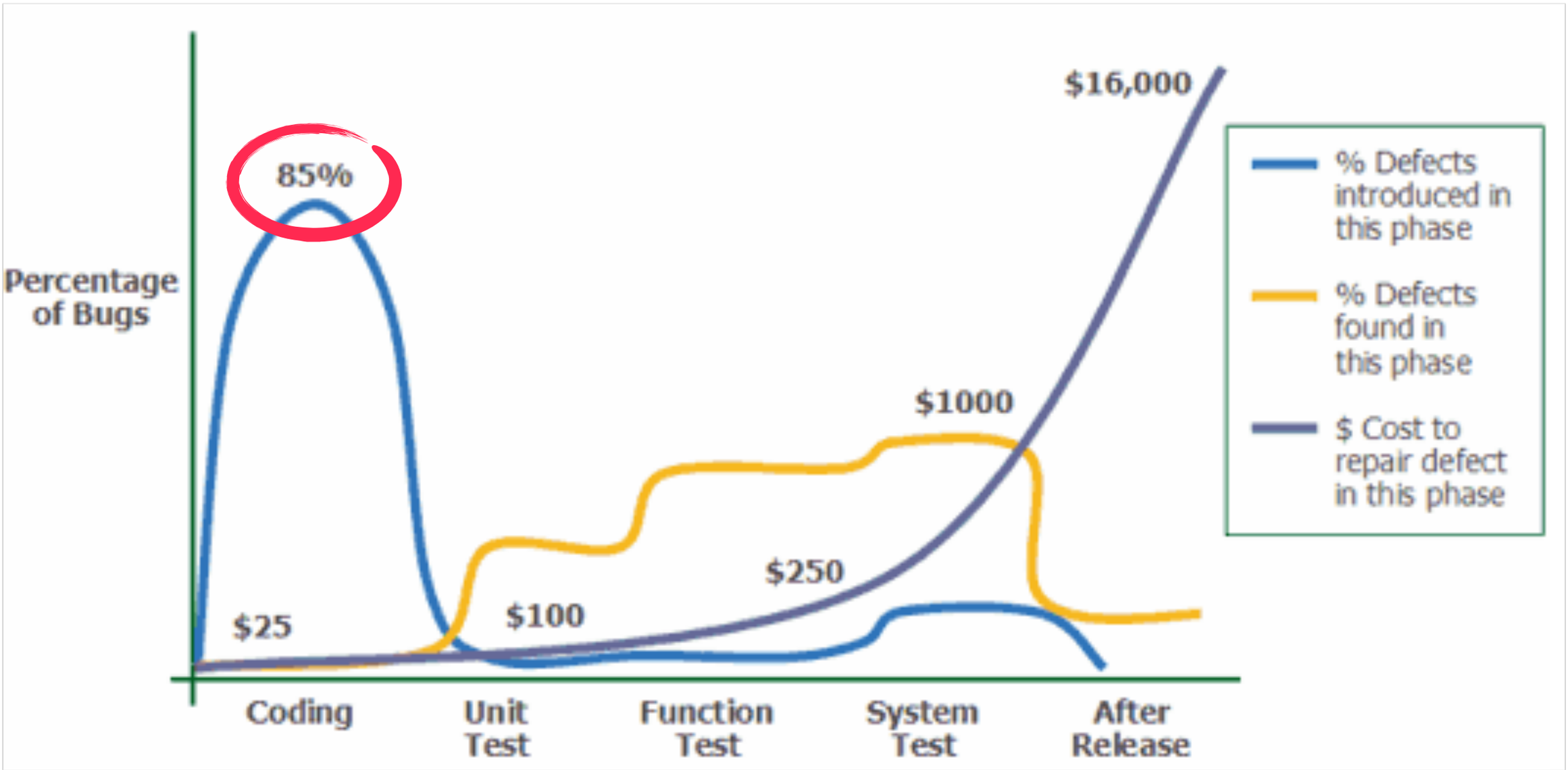
LoC : 157 k ± 23 k @ 1σ

Author present : 70% ± 14% @ 1σ

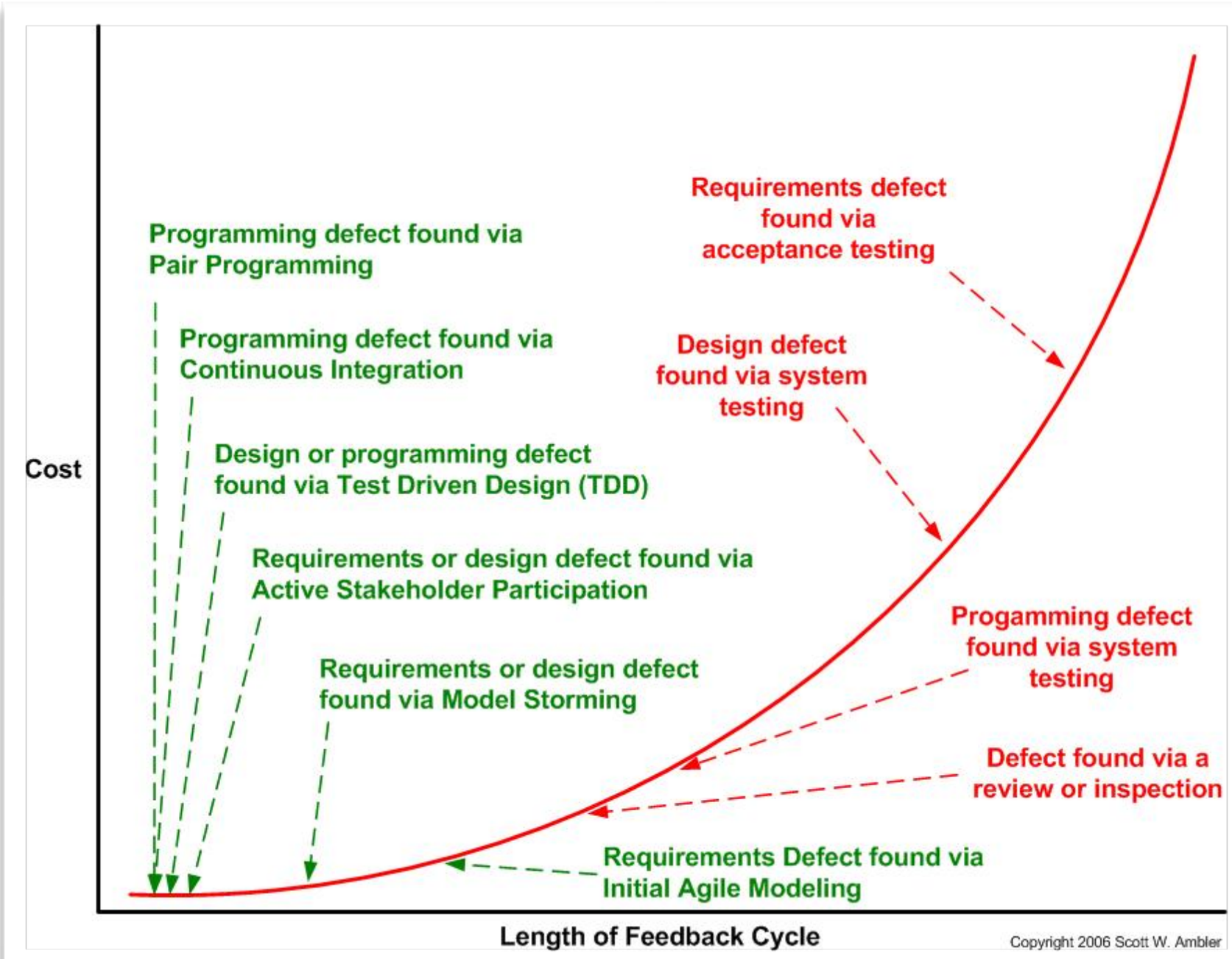

20 years
Team Size : 21
Cumulative team size : 114 ± 9 @ 1σ
1.8 MLoC
LoC : 1.8 M ± 0.08 M @ 1σ
Author present : 19% ± 4% @ 1σ

Proč testovat?

Exponenciální růst ceny opravy chyby

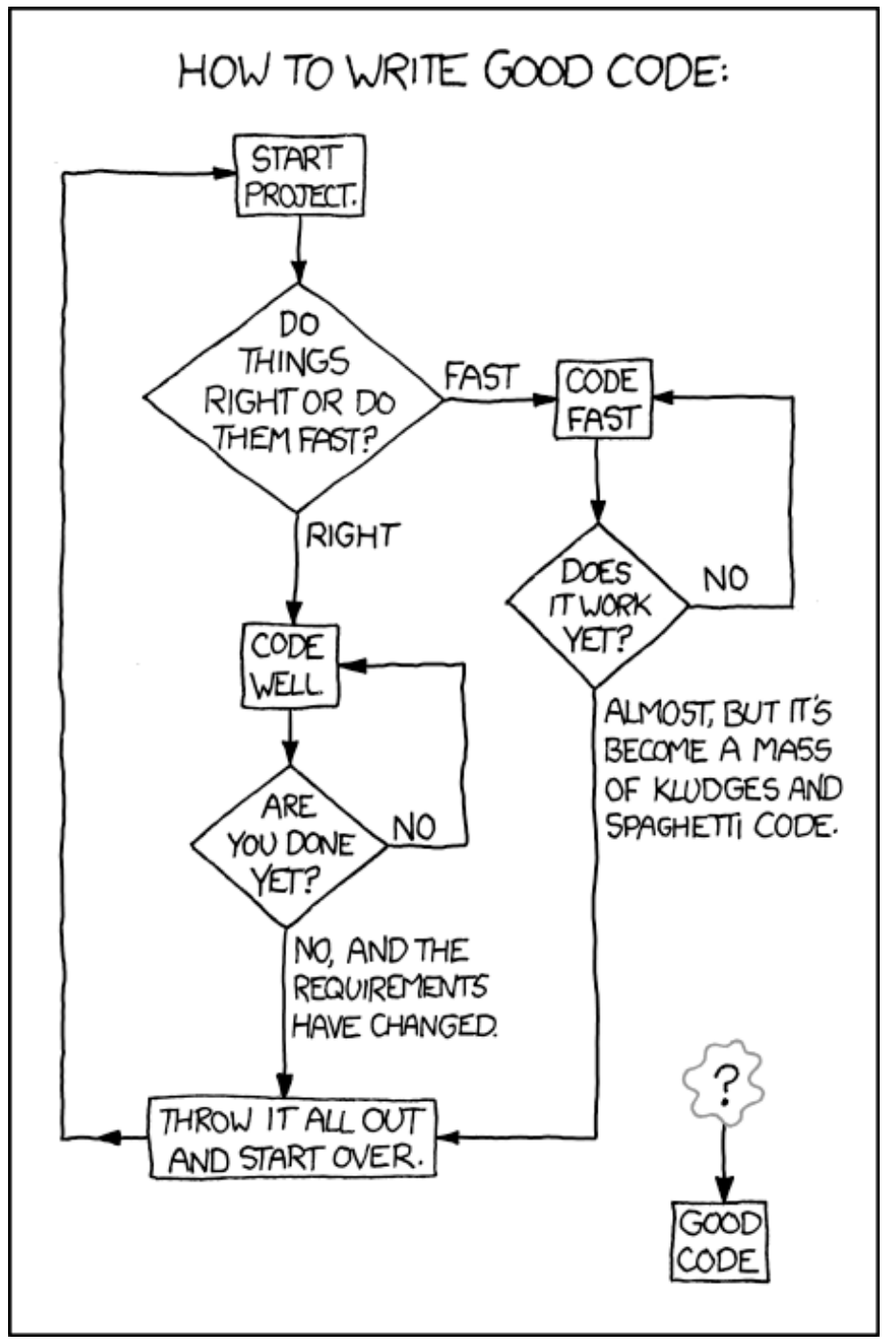


Techniky pro minimalizaci ceny opravy



Techniky pro předcházení defektů

- **Před psaním kódu**
  - komplexnost návrhu – PHP pro e-banking?
  - testovatelnost
- **V průběhu psaní kódu**
  - defenzivní programování
  - myšlení z pohledu zákazníka
  - inspekce kódu (code review)
- **Při testování kódu**
  - statická analýza kódu, profilery, …
  - mocks a stubs objekty pro testování
  - TDD



Defenzivní programování

- Inicializace všech proměnných před jejich použitím.
- Konsistentní zápis návratových hodnot
- Použití pouze jednoho bodu opuštění každé procedury/metody
    - Jak to, že nemám dostatek paměti?
    - Odkud se zde proboha bere ten null?
- Použití assert, který má smysl
    - Je tato hodnota, kterou předpokládám true, opravdu true?
- Psaní čitelného kódu
    - Jsou jména všech proměnných a metod dostatečně popisná a reprezentují dané chování či vlastnost?
    - Jsou používány dostatečné komentáře a to konzistentním způsobem (například metody komentujeme v definici rozhraní)?

```java
public Calendar getTomorrow() {
    Thread.sleep(1000*60*60*24);
    return Calendar.getInstance();
}
```

Zítra?

Jak zkontrolovat příponu souboru...

```cpp
bool checkForExe(std::string ext)
{
    return ext==".exe" ? true :
                ext==".Exe" ? true :
                ext==".eXe" ? true :
                ext==".EXe" ? true :
                ext==".exE" ? true :
                ext==".ExE" ? true :
                ext==".eXE" ? true :
                ext==".EXE" ? true : false;
}
```

```php
$int = (int) $int;
if (is_int($int)) { // pro jistotu
    // ...
}
```

PHP je silně netypové, lépe je ověřovat...

Komentáře?

```
protected void Page_Load(object sender, EventArgs e)
{
  if (!IsPostBack)
    zakazaniPolicekNaKterePrihlasenyUzivatelNemaPravoAbySiJeNemohlZaskrtnoutAPridatSiTakPravaNaKtereSamNemaPravo();
}

private void zakazaniPolicekNaKterePrihlasenyUzivatelNemaPravoAbySiJeNemohlZaskrtnoutAPridatSiTakPravaNaKtereSamNemaPravo()
{
  ...
}
```

Validátor

```
function formValidator(){
    // Check each input in the order that it appears in the form!
    if(isAlphabet(firstname)){
        if(isAlphabet(surname)){
            if(isAlphabet(surname)){
                if(ValidateDate(dob)){
                    if(isAlphabet(pgname)){
                        if(isAlphanumeric(Address1)){
                            if(isAlphanumeric(Address2)){
                                if(isAlphanumeric(Address3)){
                                    if(isAlphanumeric(Address4)){
                                        if(isAlphanumeric(Address5)){
                                            if(emailValidator(Email)){
                                                if(PhoneValidator(lan)){
                                                    if(PhoneValidator(mobile)){
                                                        if(Sessions()){
                                                            return true;
                                                        }
                                                    }
                                                }
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
    return false;
}
```


EVERY TIME YOU WRITE CRAPPY CODE
GOD KILLS A KITTEN

Implementation podle RUP

- Mapování návrhu na kód – popisuje způsob transformace návrhových modelů do zdrojového (spustitelného) kódu. Může se lišit podle použité technologie či architektury.

  Techniky:

- Buildování a neustálá integrace, testování
- TDD, refaktoring,
- Párové programování
- Konvence zápisu kódu
- Využívání mechanismů architektury

3 způsoby tvorby kódu

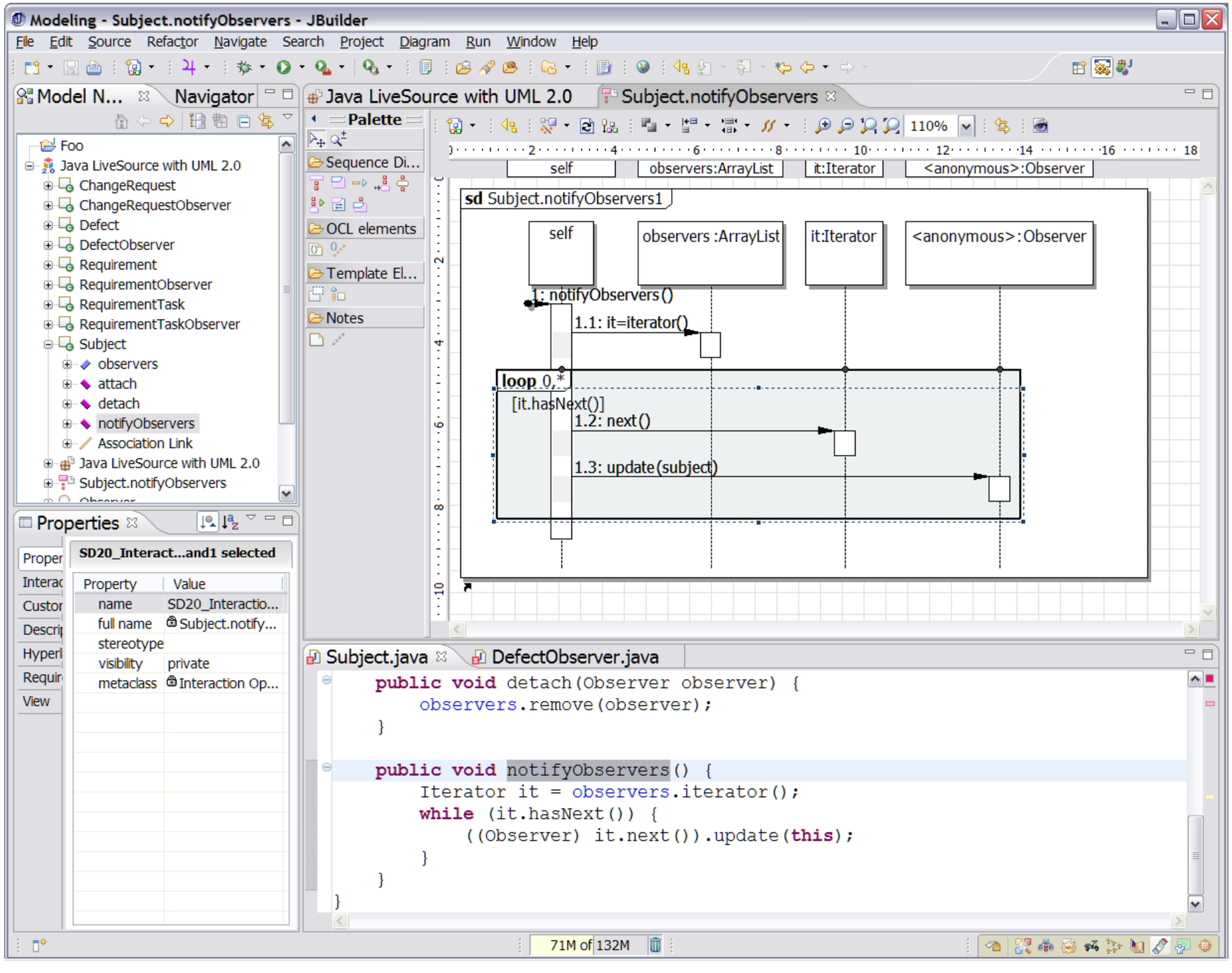
A. Vysokoúrovňový návrhový model – detaily v kódu

B. Round trip engineering – detailní modely, kód generován automaticky nástroji (také MDD)

C. TDD

- výběr nejvhodnějšího implementačního řešení.
- ihned po napsání zdrojového kódu jsme schopni ověřit funkčnost kódu unitovým testem.
- máme okamžitou zpětnou vazbu o funkčnosti a kvalitě kódu.
- v případě refaktoringu, implementace změny či rozšíření okamžitě vidíme, jestli jsme nezanesli do programu chybu.

2. možnost


Modeling - Subject.notifyObservers - JBuilder
File Edit Source Refactor Navigate Search Project Diagram Run Window Help
Model N... Navigator
Java LiveSource with UML 2.0 Subject.notifyObservers
Palette
Sequence Di...
OCL elements
Template El...
Notes
Foo
Java LiveSource with UML 2.0
ChangeRequest
ChangeRequestObserver
Defect
DefectObserver
Requirement
RequirementObserver
RequirementTask
RequirementTaskObserver
Subject
observers
attach
detach
notifyObservers
Association Link
Java LiveSource with UML 2.0
Subject.notifyObservers
Properties
SD20_Interact...and1 selected
Property Value
name SD20_Interactio...
full name Subject.notify...
stereotype
visibility private
metaclass Interaction Op...
110%
self
observers:ArrayList
it:Iterator
<anonymous>:Observer
sd Subject.notifyObservers1
1: notifyObservers()
1.1: it=iterator()
loop 0,*
[it.hasNext()]
1.2: next()
1.3: update(subject)
Subject.java
DefectObserver.java
public void detach(Observer observer) {
observers.remove(observer);
}
public void notifyObservers() {
Iterator it = observers.iterator();
while (it.hasNext()) {
((Observer) it.next()).update(this);
}
}
}
71M of 132M

3. možnost

1. Vytvoříme nový test – test nemůže projít, neexistuje implementace, kterou má testovat; důležité je mít pochopení problematiky (požadavků) např. z detailních Use Casů.

2. Spuštění všech unit testů a zjištění, že nový nebyl úspěšný – ověření, že nový nemůže projít, jelikož ještě neexistuje kód, který má testovat; ověření jeho správnosti (chyby v testu).

3. Vytvoření nějakého kódu – napsání kódu za účelem úspěšného průchodu testem, nejedná se o finální implementaci!

4. Spuštění automatických testů a zjištění úspěšného průchodu novým testem – kód splňuje požadavky, můžeme začít psát finální implementaci.

5. Refaktoring kódu – zlepšení kvality kódu, jeho vyčištění, doplnění funkcionality, nahrazení kouzelných čísel či řetězců v kódu, zanesení chyby je kontrolováno spuštěním automatických testů.

6. Opakování postupu – iterativně tento postup opakujeme, jak přidáváme nové funkčnosti.

Role Developer

Continuous integration

Martin Fowler a Matthew Foemmel definují neustálou integraci jako plně automatizované, opakované buildování, které zahrnuje také testování a je spouštěno několikrát denně. To umožňuje vývojářům denně integrovat svou práci a předejít tak integračním problémům či jejich dopad redukovat a hlavně získat rychlou zpětnou vazbu o kvalitě své práce.

CI/CD

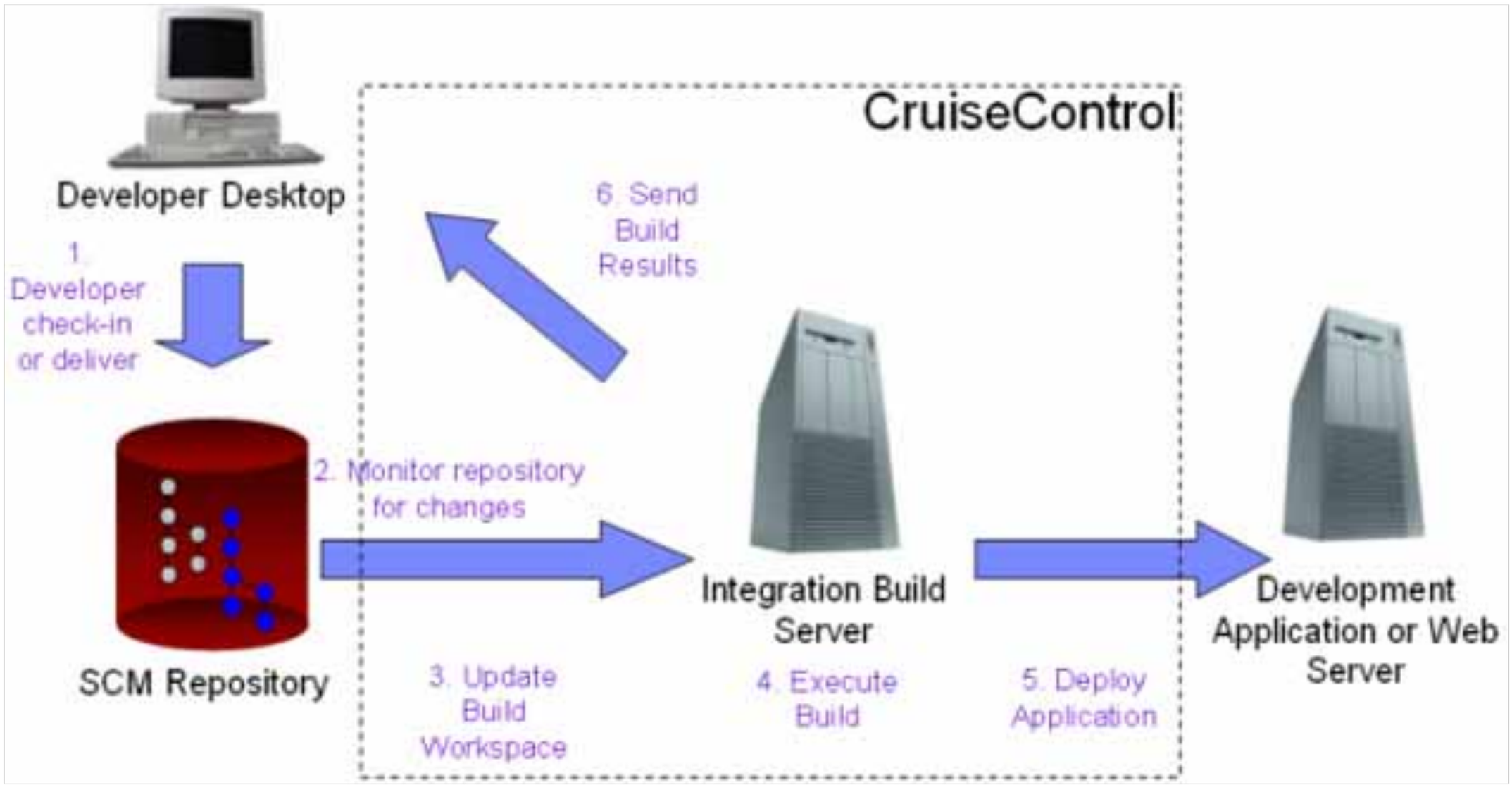
- CI - každý člen týmu průběžně integruje svůj výsledek (většinou denně)

- Continuous Delivery - Snažíte se sestavovat software tak, aby v každé chvíli byl doručitelný do produkce

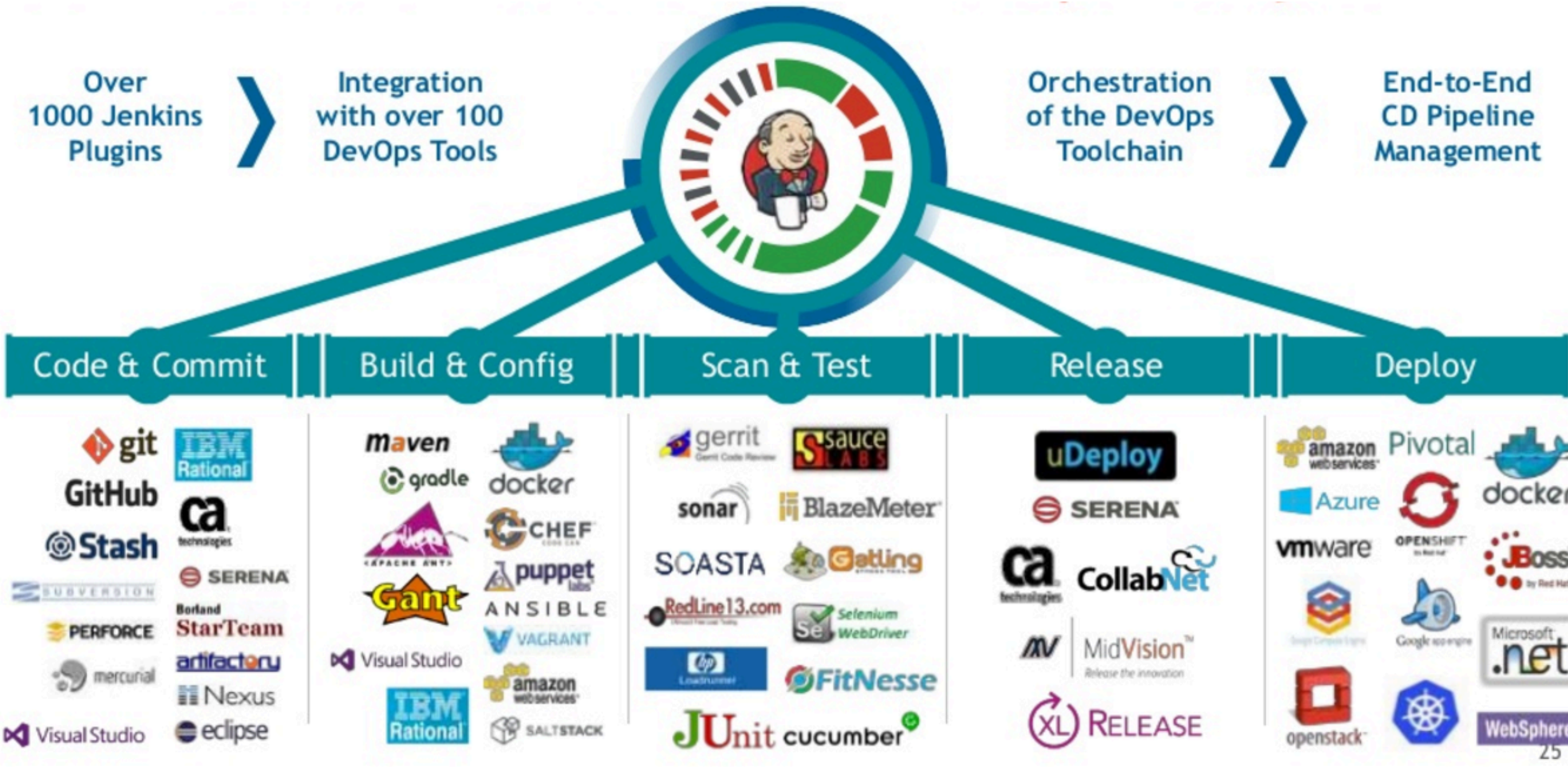
- Continuous Deployment - Rozšiřuje Continuous Delivery o automatizovaný deploy do produkce, zavádí tzv. deployment pipeline

Mapa pro orientaci v CD

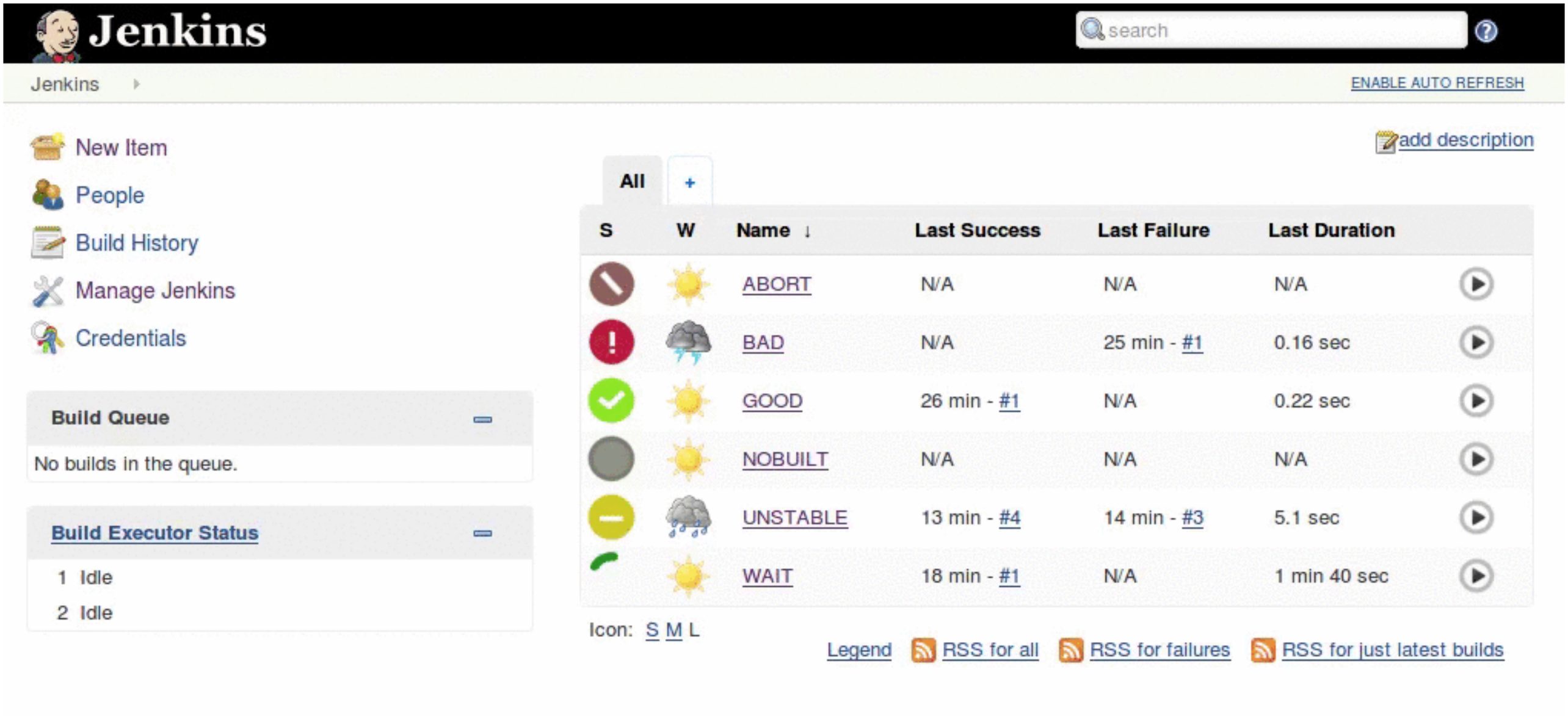
Continuous integration



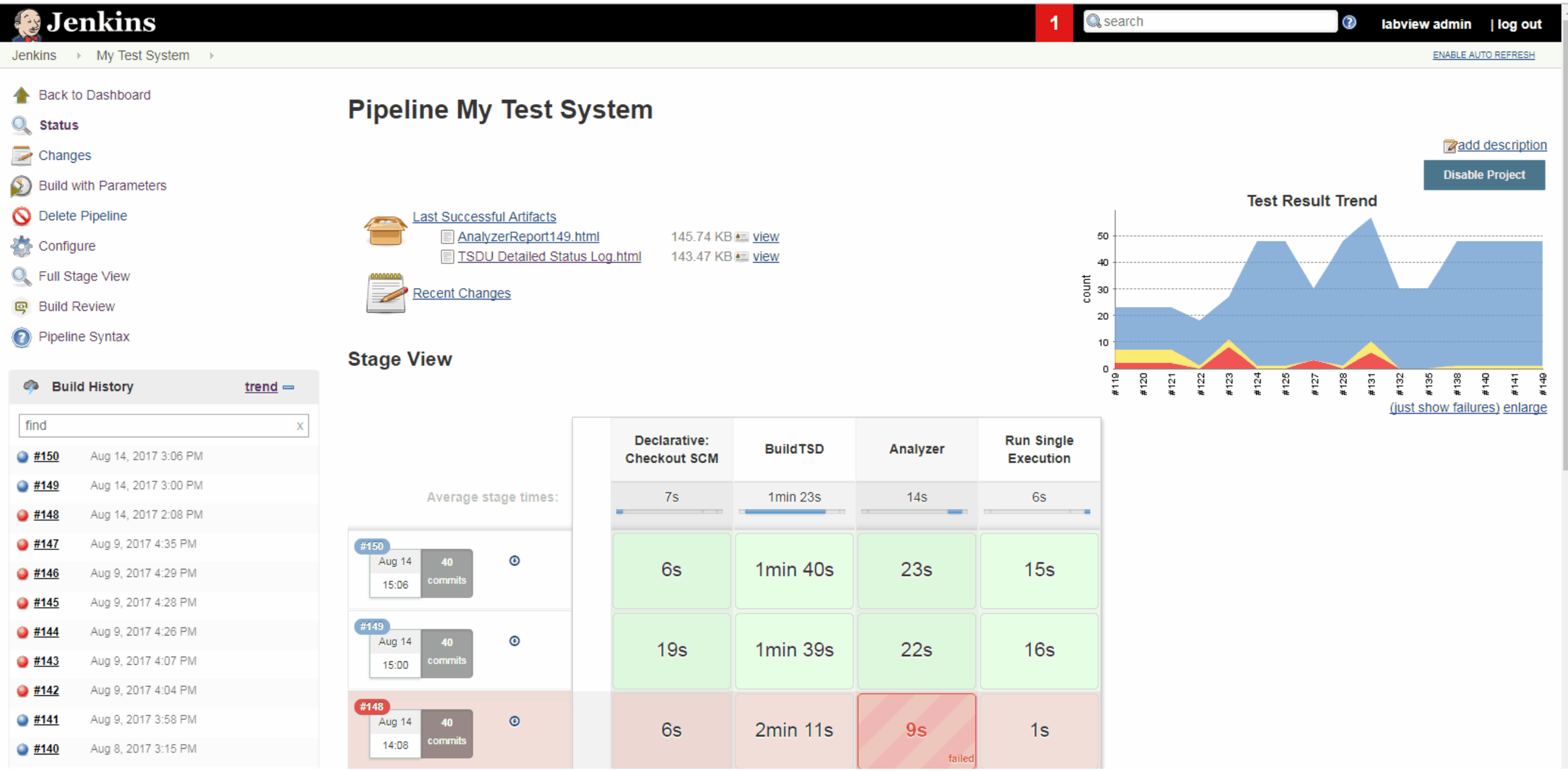
Jenkins


Over 1000 Jenkins Plugins
Integration with over 100 DevOps Tools
Orchestration of the DevOps Toolchain
End-to-End CD Pipeline Management
Code & Commit
Build & Config
Scan & Test
Release
Deploy
git
GitHub
Stash
SUBVERSION
PERFORCE
mercurial
Visual Studio
IBM Rational
ca technologies
SERENA
Borland StarTeam
artifactory
Nexus
eclipse
maven
gradle
<APACHE ANT>
Gant
Visual Studio
IBM Rational
docker
CHEF
puppet labs
ANSIBLE
VAGRANT
amazon web services
SALTSTACK
gerrit
Gerrit Code Review
Sauce LABS
sonar
BlazeMeter
SOASTA
Gatling
RedLine13.com
Selenium WebDriver
Loadrunner
FitNesse
JUnit
cucumber
uDeploy
SERENA
ca technologies
CollabNet
MidVision
Release the innovation
XL RELEASE
amazon web services
Pivotal
Azure
vmware
OPENSHIFT
docker
JBoss by Red Hat
Google app engine
Microsoft .net
openstack
WebSphere.

Jenkins


Jenkins
search
Jenkins
ENABLE AUTO REFRESH
New Item
People
Build History
Manage Jenkins
Credentials
add description
All
+
S
W
Name ↓
Last Success
Last Failure
Last Duration
ABORT
N/A
N/A
N/A
BAD
N/A
25 min - #1
0.16 sec
GOOD
26 min - #1
N/A
0.22 sec
NOBUILT
N/A
N/A
N/A
UNSTABLE
13 min - #4
14 min - #3
5.1 sec
WAIT
18 min - #1
N/A
1 min 40 sec
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Icon: S M L
Legend
RSS for all
RSS for failures
RSS for just latest builds

Jenkins


Jenkins
1
search
labview admin
| log out
Jenkins
My Test System
ENABLE AUTO REFRESH
Back to Dashboard
Status
Changes
Build with Parameters
Delete Pipeline
Configure
Full Stage View
Build Review
Pipeline Syntax
Build History
trend
find
#150 Aug 14, 2017 3:06 PM
#149 Aug 14, 2017 3:00 PM
#148 Aug 14, 2017 2:08 PM
#147 Aug 9, 2017 4:35 PM
#146 Aug 9, 2017 4:29 PM
#145 Aug 9, 2017 4:28 PM
#144 Aug 9, 2017 4:26 PM
#143 Aug 9, 2017 4:07 PM
#142 Aug 9, 2017 4:04 PM
#141 Aug 9, 2017 3:58 PM
#140 Aug 8, 2017 3:15 PM
Pipeline My Test System
add description
Disable Project
Last Successful Artifacts
AnalyzerReport149.html 145.74 KB view
TSDU Detailed Status Log.html 143.47 KB view
Recent Changes
Test Result Trend
count
(just show failures) enlarge
Stage View
Declarative: Checkout SCM
BuildTSD
Analyzer
Run Single Execution
Average stage times:
7s
1min 23s
14s
6s
#150 Aug 14 15:06 40 commits
6s
1min 40s
23s
15s
#149 Aug 14 15:00 40 commits
19s
1min 39s
22s
16s
#148 Aug 14 14:08 40 commits
6s
2min 11s
9s
failed
1s

https://oncoscape.sttrcancer.org

GitHub

Fredhutch/Oncoscape

Master

Develop

Feature XYZ

Merge

Pull Request

Trigger Pipeline

Test Results

Fork

FredHutch

Feature XYZ

Git

STTR Developers

Feature ABC

Git

External Developers

circleci

Build : all branches

Pull

Build

Test: all branches

Run

Connect

Test

Pass?

Push: master / develop

Push Image

Deploy: master/develop

Trigger Deploy

Push/Register Images

Trigger Redeployment

docker Hub

Fredhutch/Oncoscape

Latest (master)

Develop

docker Cloud

Oncoscape Prod Service

Oncoscape Develop Service

Report Events

slack

Fredhutch-STTR

#oncoscape channel

Build/Test/Push/Deploy Results

Notify Events/Status

Redeploy

AWS

amazon web services

Load Balancer

PROD

Oncoscape-prod-a

Oncoscape-prod-b

Availability Zone A

Availability Zone B

Redeploy

https://dev.oncoscape.sttrcancer.io

AWS

amazon web services

Load Balancer

DEV

Oncoscape-dev-a

Oncoscape-dev-b

Availability Zone A

Availability Zone B

Refaktoring

- Zásah do programu, jenž zlepšuje čitelnost kódu nebo jeho strukturu bez změny chování programu.

- Nedílnou součástí refaktoringu je unitové testování zajišťující zpětnou kontrolu.

- Refaktoring neodstraňuje chyby, ani nepřidává novou funkcionalitu, cílem je zlepšit čitelnost kódu, jeho strukturu, zlepšit návrh, odstranit části kódu, které se již nevyužívají.

- Neděláme nárazově pro celý projekt, ale postupně

Refaktoring

Co refaktorovat:

- úpravy metod,
- přesouvání elementů mezi objekty,
- organizace dat,
- zjednodušení podmíněných příkazů,
- zjednodušení volání metod,
- generalizace

Symptomy:

- funkční třídy (příliš mnoho odpovědností/metod v 1 třídě),
- duplicitní kód,
- dlouhé metody a seznamy parametrů,

Nástroje pro refaktoring

invitation.js

```
1   import React from 'react';
2   import ReactDOM from 'react-dom';
3   import {HelloMessage} from './hello';
4
5   class Invitation extends React.Component {...}
31
32  Invitation.propTypes = { name: React.PropTypes.string };
33  Invitation.defaultProps = { name: "" };
34
35  ReactDOM.render(<Invitation/>, document.querySelector('.container'));
36
```

Příklady refaktoringu

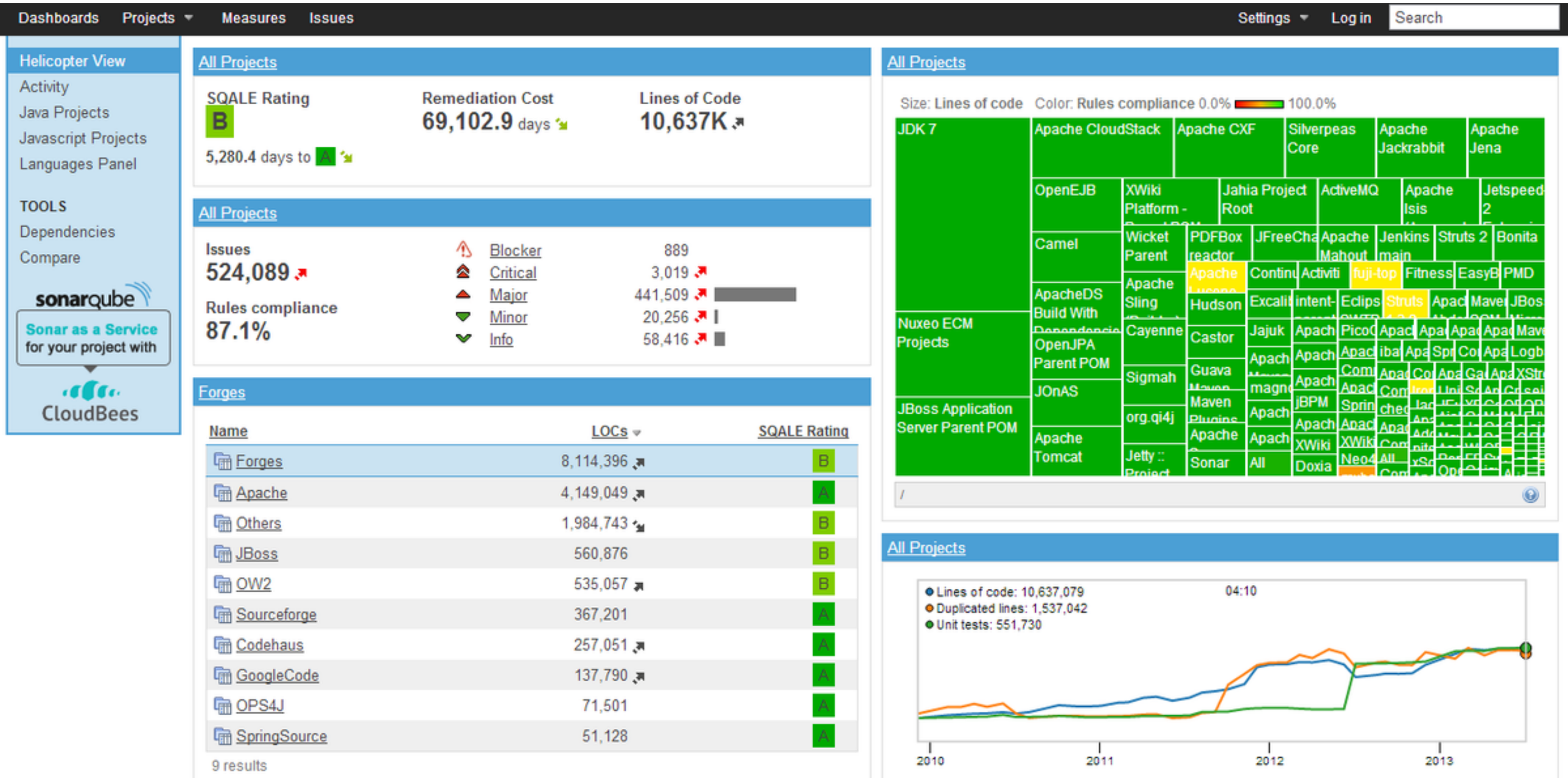
```
3 references
export class TypeScriptVersionPicker {
    6 references
    private _currentVersion: TypeScriptVersion;

    1 reference
    public constructor(…
    ) {…
    }

    1 reference
    public async show(firstRun?: boolean): Promise<{ oldVersion?: TypeScriptVersion, newVersion?:
        const pickOptions: MyQuickPickItem[] = [];

        pickOptions.push({
            label: localize('learnMore', 'Learn More'),
            description: '',
            id: MessageAction.learnMore
        });

        const shippedVersion = this.versionProvider.defaultVersion;

        pickOptions.push({
            label: (!this.useWorkspaceTsdkSetting
```

99 Bottles part 1

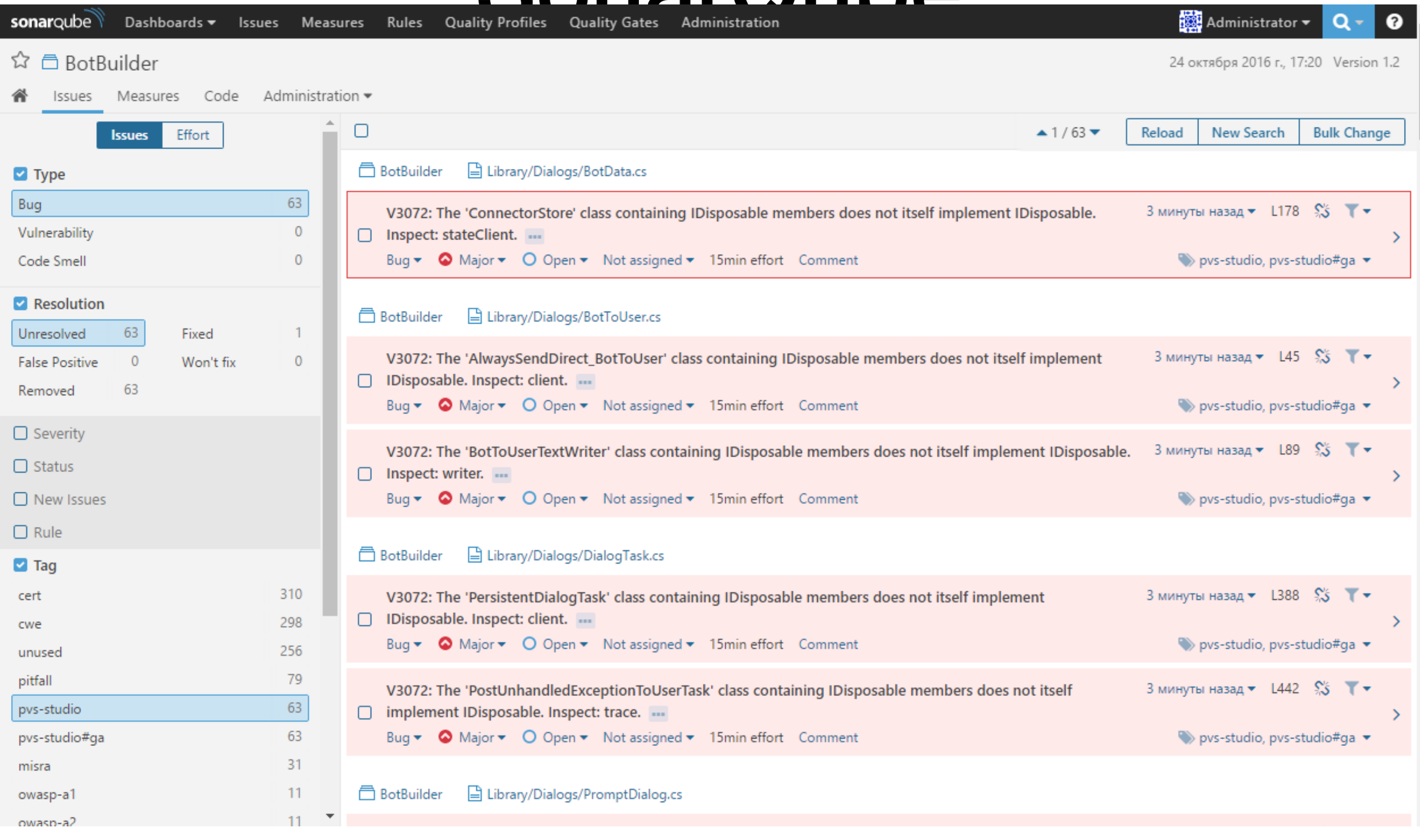
99 Bottles part 2

99 Bottles part 3

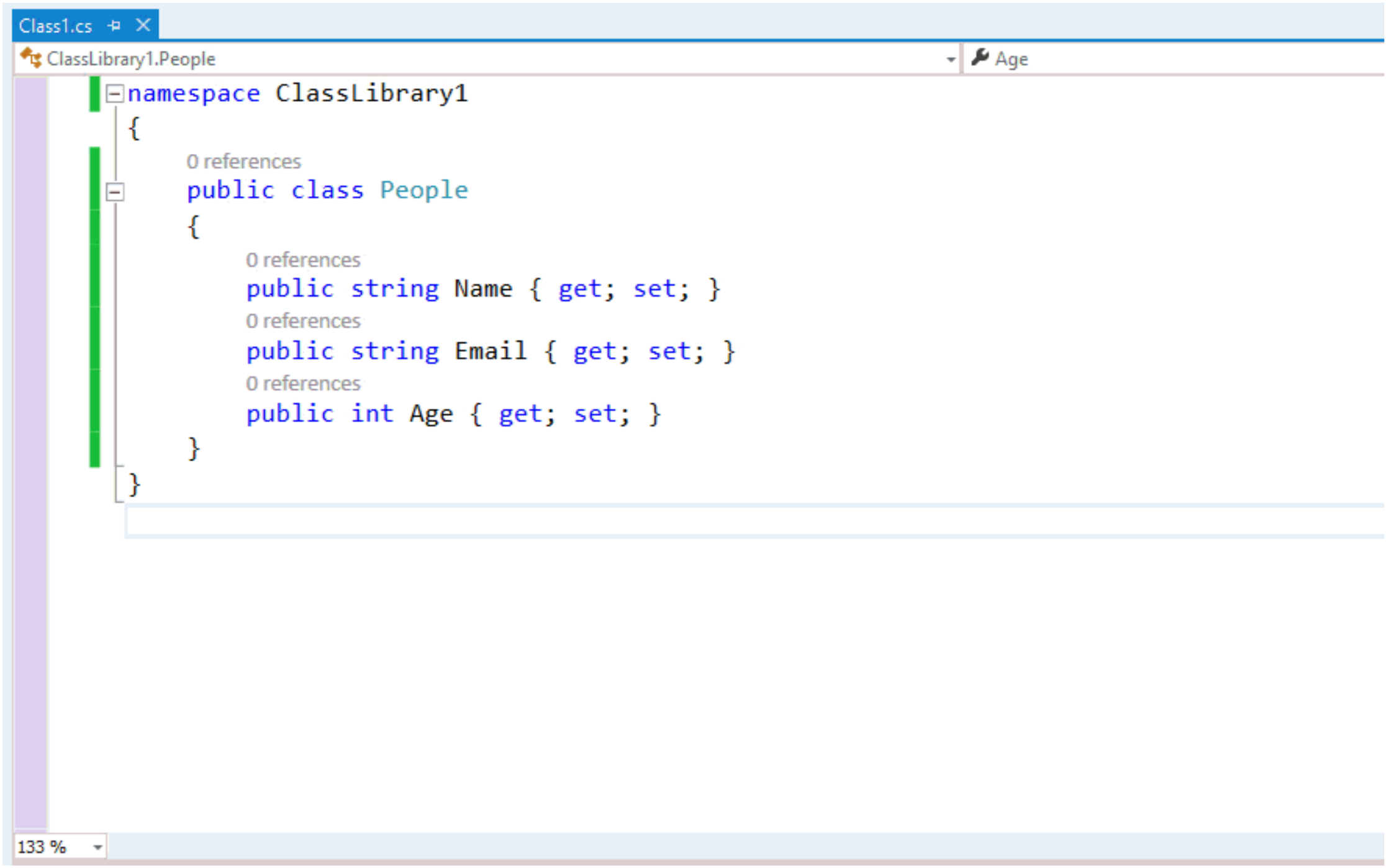
Nástroje pro udržení kvality kódu - SonarQube



Typický výstup ze SonarQube



Nástroje pro udržení kvality kódu ReSharper



Statická analýza kódu

- Nástroje pro kontrolu kódu programu bez jeho spuštění.
- Automatizovaně vyhledává a označuje programátorské a stylistické chyby.
- Každý jazyk má své vlastní pravidla kontroly a konvence, lze je rozšiřovat a přepisovat.
- Realizován nástrojem linter (pro příslušný jazyk).

lint-test.ts

```ts
const foo = 'foo';
const bar = "bar";
const baz: number = 5;

let quux: any;

quux = 5;

let arr = [
    'foo',
    'bar',
    'baz',
];

for (const current in arr) {
    console.log(current);
}

function func(arg1: any, arg2) {
    console.log(arg1);
    console.log(arg2);
}

(async () => {
    console.log('async');
})();

class MyClass {
    log(): void {
        console.log(this);
    }
}

const instance = new MyClass();
const log = instance.log;

log();
```

Párové programování

- Zlepšené programování – děláme správnou věc.
- Větší kvalita kódu – před kolegou programátorem tíhneme k tomu přijít s lepším řešením, než bychom použili, pokud bychom kód psali sami.
- Vícero vývojářů přispívá k návrhu aplikace – při rotaci párů se u kódu vystřídá více programátorů, jejich názory/řešení jsou konfrontovány s více lidmi.
- Zvýšená morálka – pro některé je programování v páru zábavnější než programování o samotě.
- Kolektivní vlastnictví kódu – pokud každý na projektu párově programuje a páry často rotují, každý člen týmu si vytvoří znalost zahrnující celou bázi kódu, nejen části aplikace.
- Mentoring – párové programování je nejjednodušší (rozuměj nejméně bolestná) cesta předání znalosti.
- Větší sounáležitost týmu – lidé se více znají, střídají se ve spolupráci.
- Programátoři jsou méně vyrušováni.

Kde je vhodné ho použít?

- Revize kódu
    - Vysvětlení přísedícímu = může si všimnout nějaké chyby či nevhodné konstrukce.
    - Vyčleníme určitý časový úsek denně.
- Spolupráce zkušeného a nezkušeného programátora
    - nezkušený se učí tím, že okoukává práci zkušeného
    - nezkušený sedí u klávesnice, zkušený na něj dohlíží, vysvětluje a instruuje ho, v případě potřeby se vymění, aby bylo vysvětlení prakticky ukázáno.
- Předání znalosti architektury (mechanismů architektury)
    - architekt párově programuje s programátory (ukázka použití mechanismů architektury) – cílem je udržení kvalitního a přehledného kódu.

Návrhové vzory

- V Construction refaktorujeme jednoduchý kód z Elaboration (využití DP)
- Lépe strukturovaný kód
- Lepší čitelnost
- Lepší rozšiřitelnost (změny na 1 místě)
- Použití Factory, Singleton, Interceptor, Proxy, Pool, Library, Observer, …

Prototypy

- Start disciplíny v Inception (je-li třeba prototyp)
- Demonstrace technologie, určitého řešení
- Spojení s legacy systémem, s DB
- Publikování reportů či XML dat

- Prototypy tvoříme v případě zjištění/ověření:
  - zda je produkt životaschopný na trhu,
  - stability či výkonnosti klíčové technologie,
  - pochopení požadavků,
  - vzhled a použitelnost produktu.

Druhy prototypů

Podle zaměření prototypu (co prozkoumává)

- prototypy chování (zkoumání specifického chování, algoritmu)
- strukturální prototypy (zkoumání možností architektury či technologie).

Podle toho, jak tento prototyp dále přežívá:

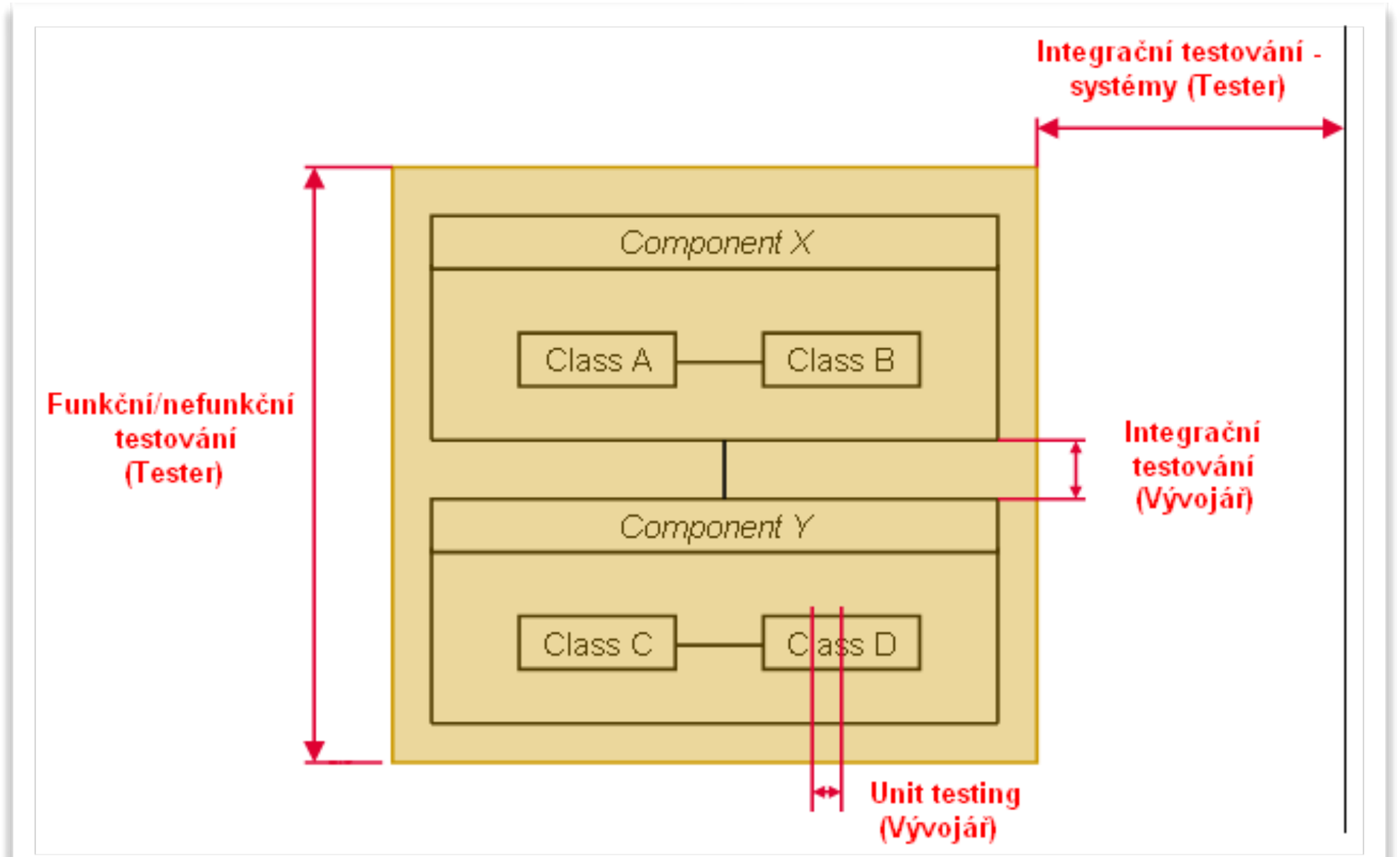
- evoluční (jsou nadále rozvíjeny a stává se z nich finální systém)
- ověřovací (pouze ověření myšlenky a poté jsou zahozeny)

Testing

Testing

- Poskytovatel služeb pro ostatní disciplíny.
- Testování zaměřeno na ověřování a hodnocení kvality jednotlivých produktů:
  - Nalezení a dokumentace defektů v kvalitě software.
  - Sdělování očekávané kvality.
  - Validace a ověření předpokladů vytvořených v návrhu a specifikaci požadavků formou konkrétní demonstrace.
  - Validace softwarového produktu podle návrhu.
  - Ověření správné implementace požadavků zákazníka.

Předmět testování



Druhy testování

- **Unitové** – zaměřuje se na ověření nejmenších testovatelných jednotek systému (typicky komponenty), probíhá v rámci implementace (disciplína Implementace).

- **Integrační** – komponenty zahrnuté do implementačního modelu (balíček – package nebo subsystém) pracují a komunikují správně při provádění daného use case?

    - Nedostatečná integrace bývá častým zdrojem problémů a selhání SW systémů.

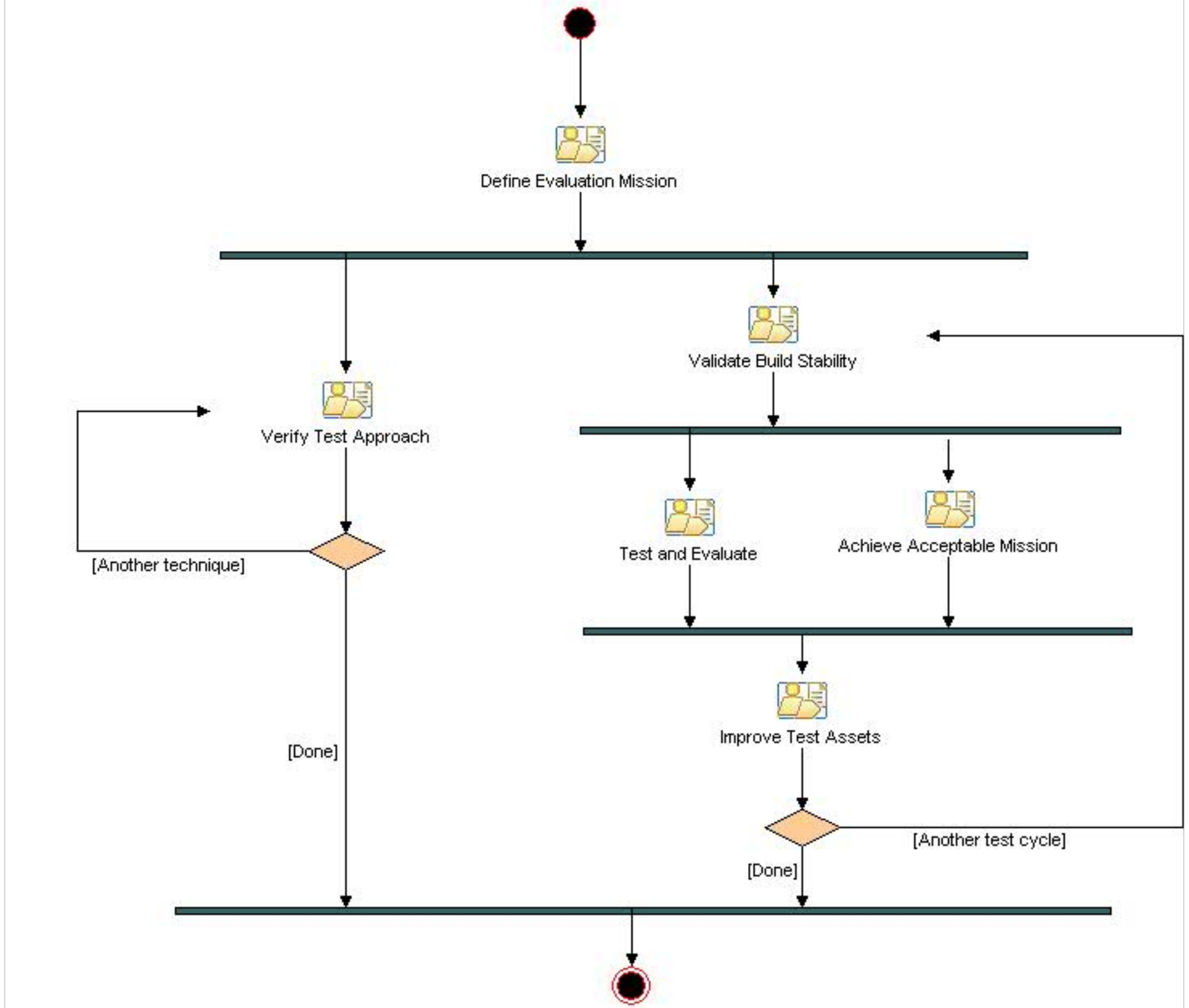
- **Systémové** – testování funkčnosti aplikace

- **Akceptační** – prováděno uživateli, jedná se o poslední testovací akci před vlastním nasazením software.

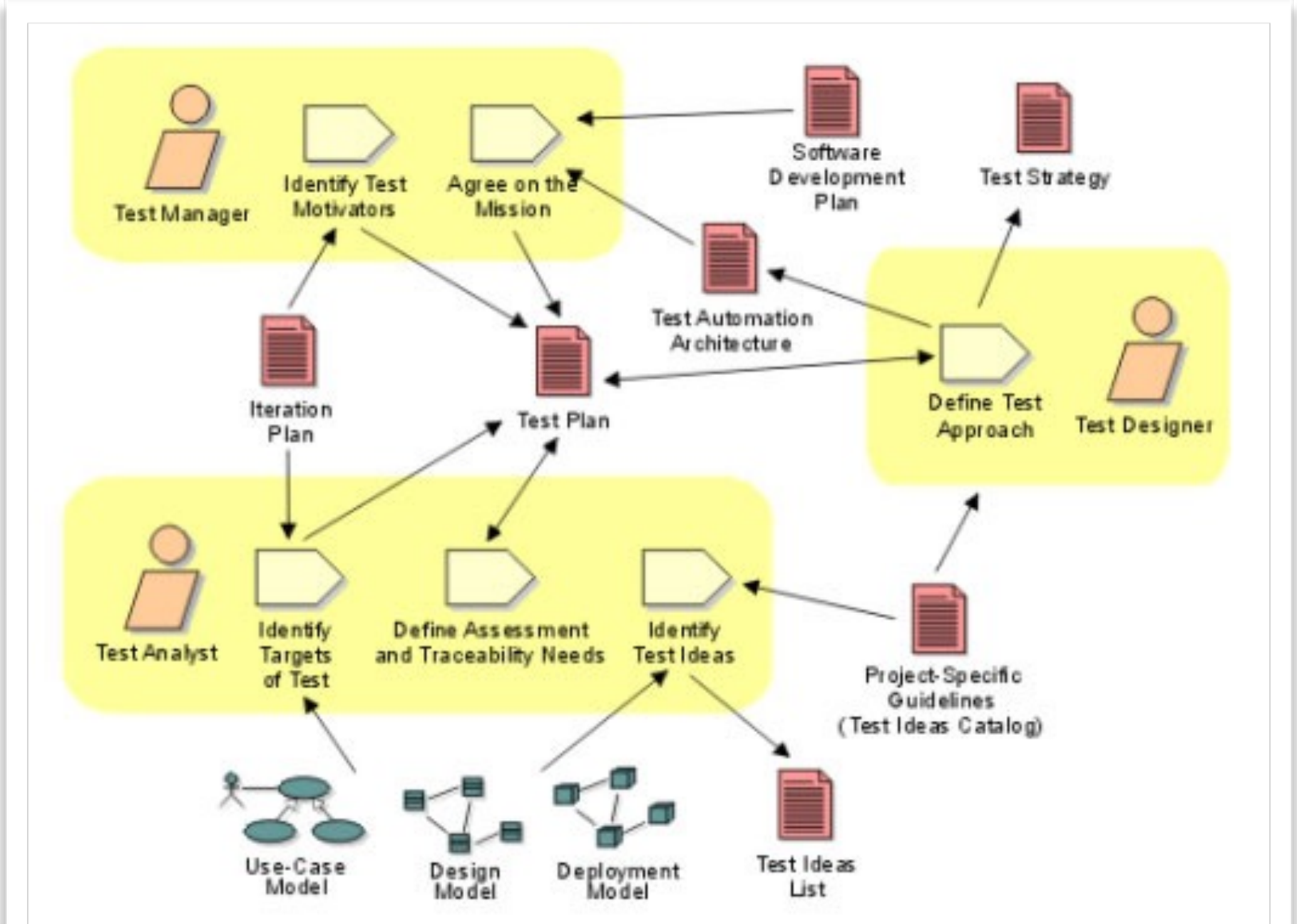
    - Cílem je ověřit, zda je systém hotový a může být používán uživateli k naplnění cílů, pro které byl vytvořen.

Aplikace F principu

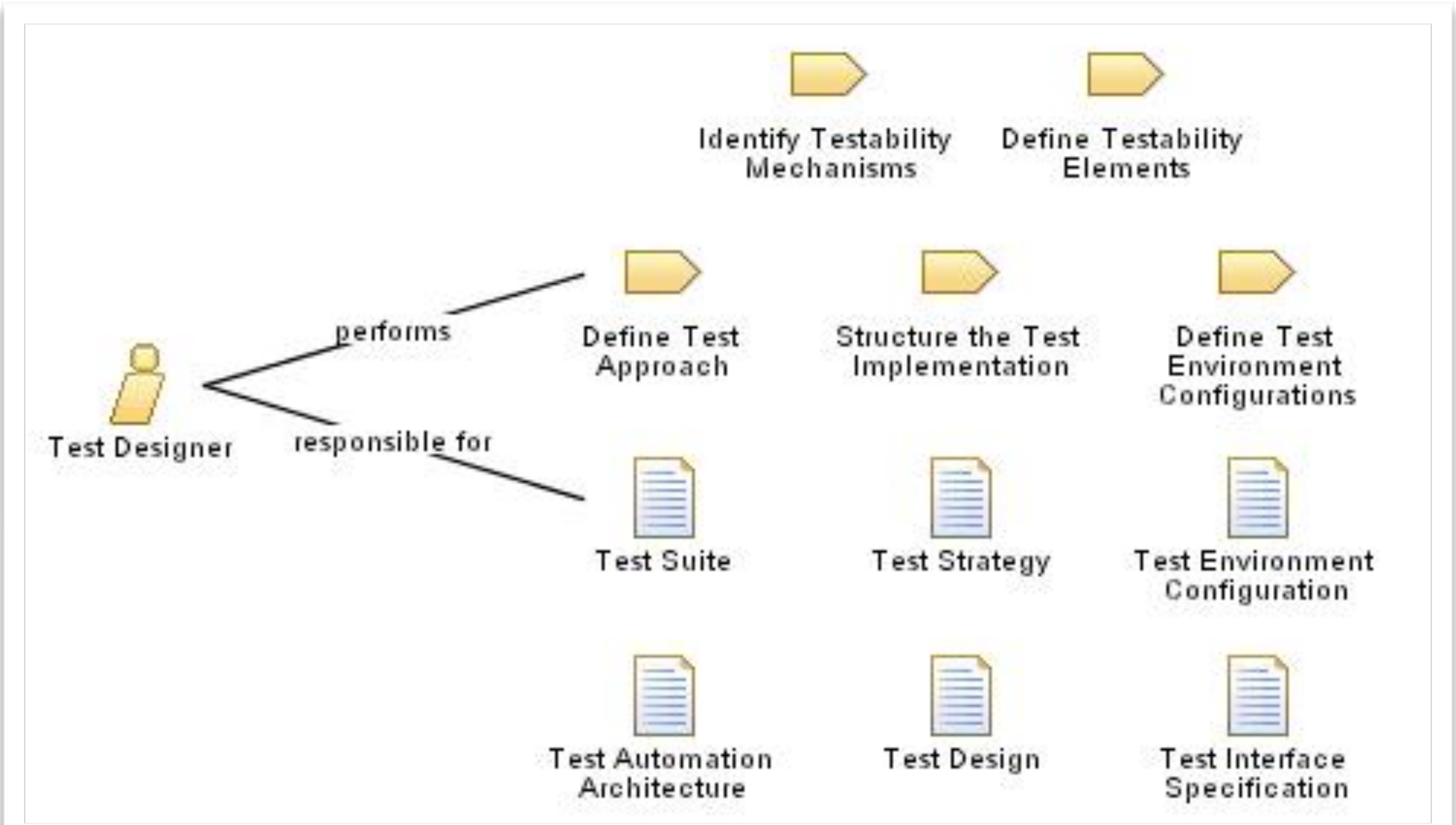
- Každý je odpovědný za kvalitu.
- Testeři nemohou korigovat špatnou práci analytiků, návrhářů a programátorů.
- Produkt musí být navrhován s ohledem na kvalitu od začátku, není možné ji „tam přidat silným tlakem managementu či zákazníka později“.
- Všechny role přispívají k výslednému kvalitnímu produktu od počátku jeho vývoje.
- Cílem testovacích rolí tedy není zajistit kvalitu, ale ohodnotit ji!
    - Dále také poskytovat častou zpětnou vazbu ostatním rolím, aby bylo možné problémy s kvalitou odstranit za rozumnou cenu v rozumném čase.


Define Evaluation Mission
Verify Test Approach
[Another technique]
[Done]
Validate Build Stability
Test and Evaluate
Achieve Acceptable Mission
Improve Test Assets
[Another test cycle]
[Done]

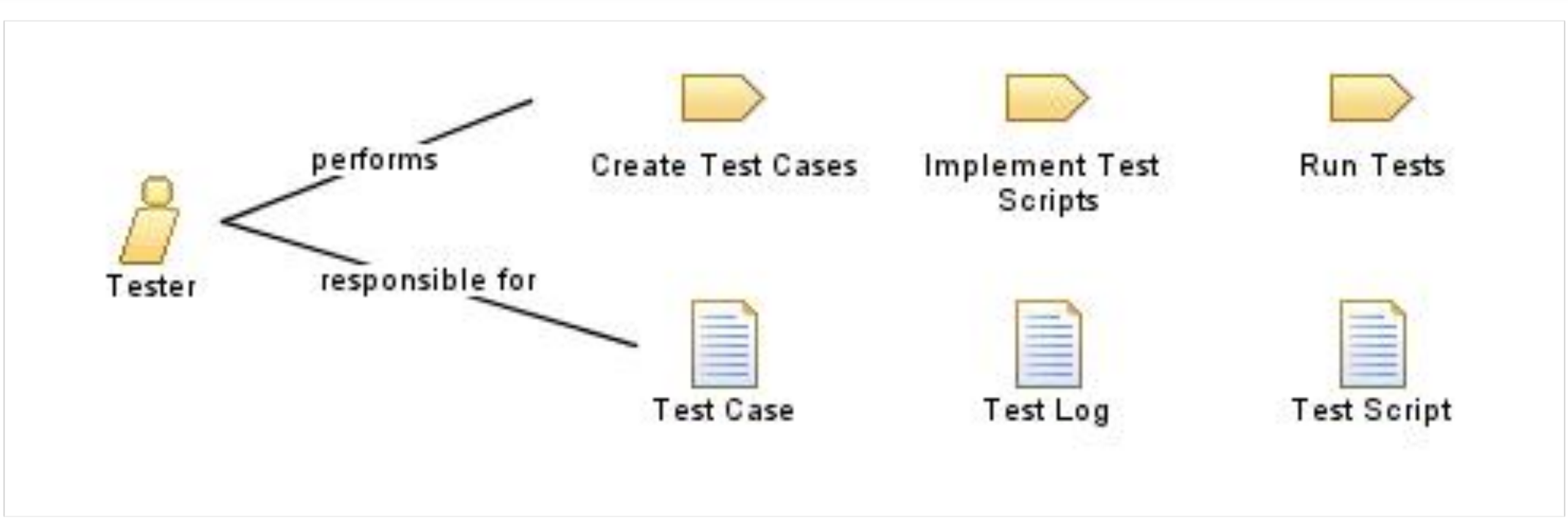
Define evaluation mission



Role Test designer



Role Tester


performs
Create Test Cases
Implement Test Scripts
Run Tests
Tester
responsible for
Test Case
Test Log
Test Script

Artefakty

- **Test case** – množina testovacích dat, podmínek a postupu provedení testu a předpokládaných výsledků testů pro specifické cíle;

- ...může být derivován z use case, návrhových dokumentů či z vlastního kódu aplikace.

- **Test skript** – počítačově zpracovatelný sled testovacích procedur, který je možné automatizovat.

- **Testovací třídy a komponenty** (mocks a stubs) – třídy a komponenty realizující navržené testy, zahrnují také ovladače a různé částečné implementace (mocks a stubs) imitující části aplikace či jiné spolupracující systémy, které ještě nejsou naprogramovány, ale je třeba je mít k dispozici k otestování naší části aplikace.

Test Case

Use Case	Scénář	Data a podmínky		
		Hodiny	Uživatel	Výstup
Reportuj čas	BF	5	user	OK
Reportuj čas	AF#1	-4	user	Chybové hlášení
...	...	...	...	...

Regresní testování

Strategie testování, dříve provedené testy provádíme opět na nové verzi aplikace s cílem zajistit, aby kvalita produktu nešla dolů, nesnížila se díky přidání nových funkčností.

Cíl:

- zajistit, že dříve odhalené chyby jsou již odstraněny,
- zajistit, že změny provedené v kódu nezanesly nové chyby či se znovu neobjevily chyby předchozí.

Test case

Vyzkoušet přihlášení a zobrazení stránky pro všechny majoritní prohlížeče

ID případu	TC_LOGIN_01
Popis	Přihlášení do systému
Aplikovatelné na	IE10, FireFox - verze 20, Chrome - verze 26, Safari - verze 6
Požadavky	Systém ověří uživatele po zadání jména a hesla a zobrazí úvodní obrazovku s monitorem záloh.
Vstupní podmínky	Do prohlížeče je vložena stránka [redacted] a je zobrazena přihlašovací stránka.

Krok	**Úkol a očekávaný výsledek**	
1	Zadání stránky do prohlížeče	
2	Zobrazení vypadá jednotně (styly, text, zalamování)	**OK /** ~~Chyba~~
3	Vložení uživatelského jména a hesla: [redacted]	

BDD

- Evoluce TDD
- Přidává doménově-řízený návrh a OOP analýzu a návrh
- Základem pro popis je doménově specifický jazyk (DSL)
- Logika je založena na test scriptech
- Nástroje se snaží o automatizaci BDD scénářů

User story template

Story Title

Narrative:
In Order to [benefit]
As a [role]
I Want to [feature]

Acceptance Criteria:
Scenario: [description]

Given [context or precondition]
When [event or action]
Then [outcome validation]

Scenario: Successful login
Given the user is on the URL /
...

```
@Given ("the user is on the URL $url")
public void goUrl(String url) {
  String formattedUrl = this.getVariableValue(url);
  String baseUrl = this.getVariableValue("@baseUrl");
  if (!formattedUrl.startsWith("http") && !baseUrl.isEmpty()) {
    formattedUrl = baseUrl + "/" + formattedUrl;
  }
  try {
    getDriver().get(formattedUrl);
  } catch (TimeoutException e) {
      if (!this.getPageLoadTimeout()) {
        throw e;
      }
  }
  this.webWait(this.getWaitAction());
}
```

Java

Cucumber

jbehave

Ruby

RSpec

Cucumber

php

behat

JS
JavaScript

Jasmine

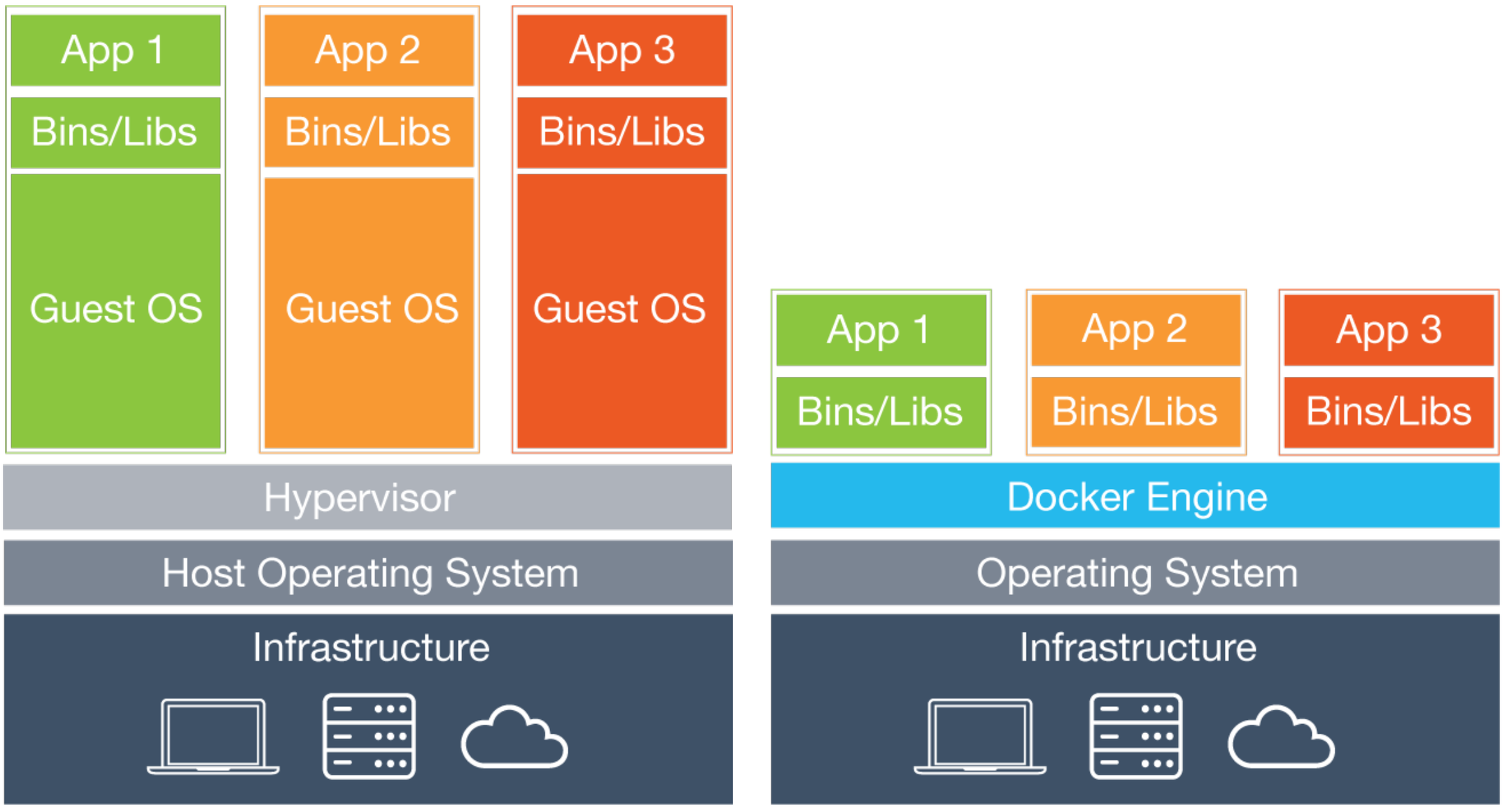
python

http://www.bddassistant.com

QA a virtualizace

- Service virtualization - na úrovni kódu/služby - stubbing/ mocking
    - SoapUI, MockServer, HP Service virualization, IBM Rational Test Virtualization Server
- Celý ekosystém
    - Hyper-V, VMware, Team Foundation Server
- Virtualizace prostředí…

Docker


App 1
Bins/Libs
Guest OS
App 2
Bins/Libs
Guest OS
App 3
Bins/Libs
Guest OS
Hypervisor
Host Operating System
Infrastructure
App 1
Bins/Libs
App 2
Bins/Libs
App 3
Bins/Libs
Docker Engine
Operating System
Infrastructure

Project management

Project management

- Příprava projektu
- Zahájení
- High level plánování

- Vykonávání
- Detailní plánování
- Vykonávání
- Řízení a monitorování

- Uzavření a zhodnocení (iterace, projektu)

Projekt

- Projekt je plánovaná, řízená, časově ohraničená skupina činností, která má dané vstupy a výstupy a spotřebovává určité zdroje (lidské, technologické, finanční).

- Definice PMBOK: projekt je dočasné úsilí s cílem vytvořit unikátní produkt nebo službu.

Trojúhelník kvality

Vztah 3 hlavních faktorů

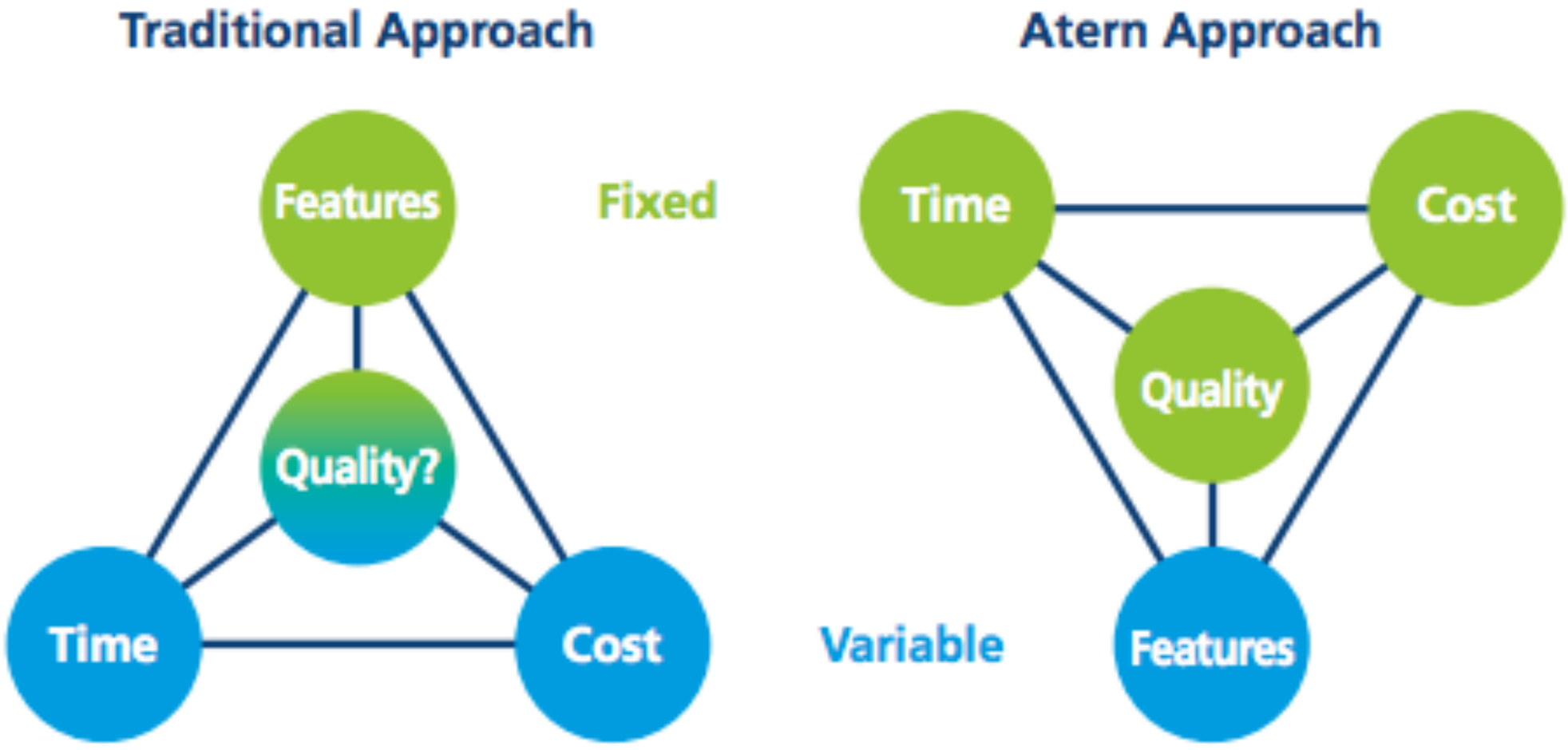
- času, nákladů a výkonu/kvality.

- Pohled zákazníka: vždy co nejkvalitnější produkt v krátkém termínu a s nízkými náklady.

- Pozor: tým si musí určit sám 1 z těchto faktorů, aby byl projekt proveditelný

  - Např. pokud bude chtít zákazník zkrátit termín, musí automaticky počítat s většími náklady nebo snížením kvality, při snaze zvýšit kvalitu je třeba navýšit náklady a/nebo prodloužit termín apod.

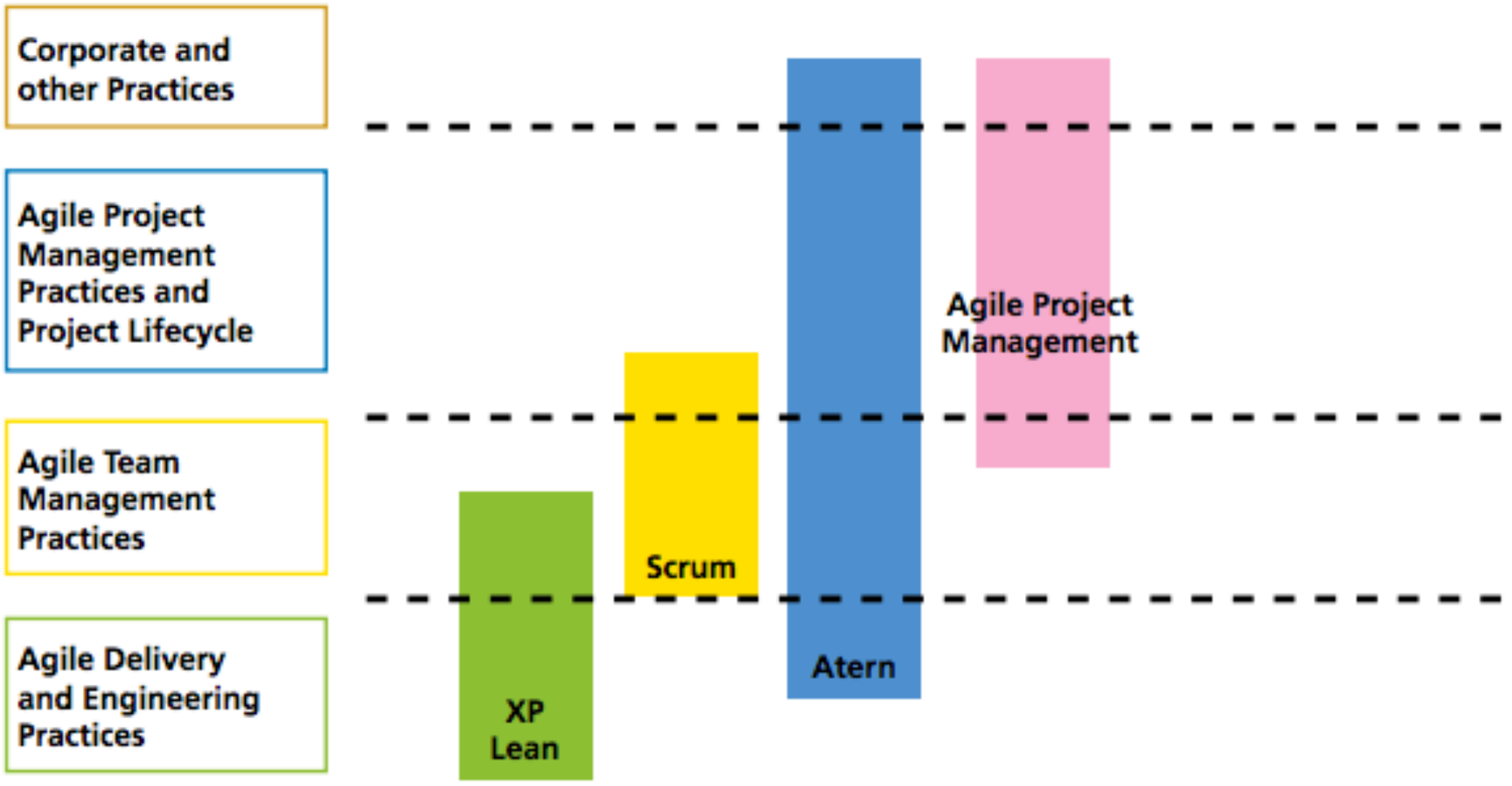
Trojúhelník kvality


Traditional Approach
Features
Fixed
Quality?
Time
Cost
Atern Approach
Time
Cost
Quality
Variable
Features

Hlavní oblasti PM

- Odhadování práce / úkolů
- Incremental funding methodology (minimum viable product)
- Řízení rizik

What the Agile Method Covers*

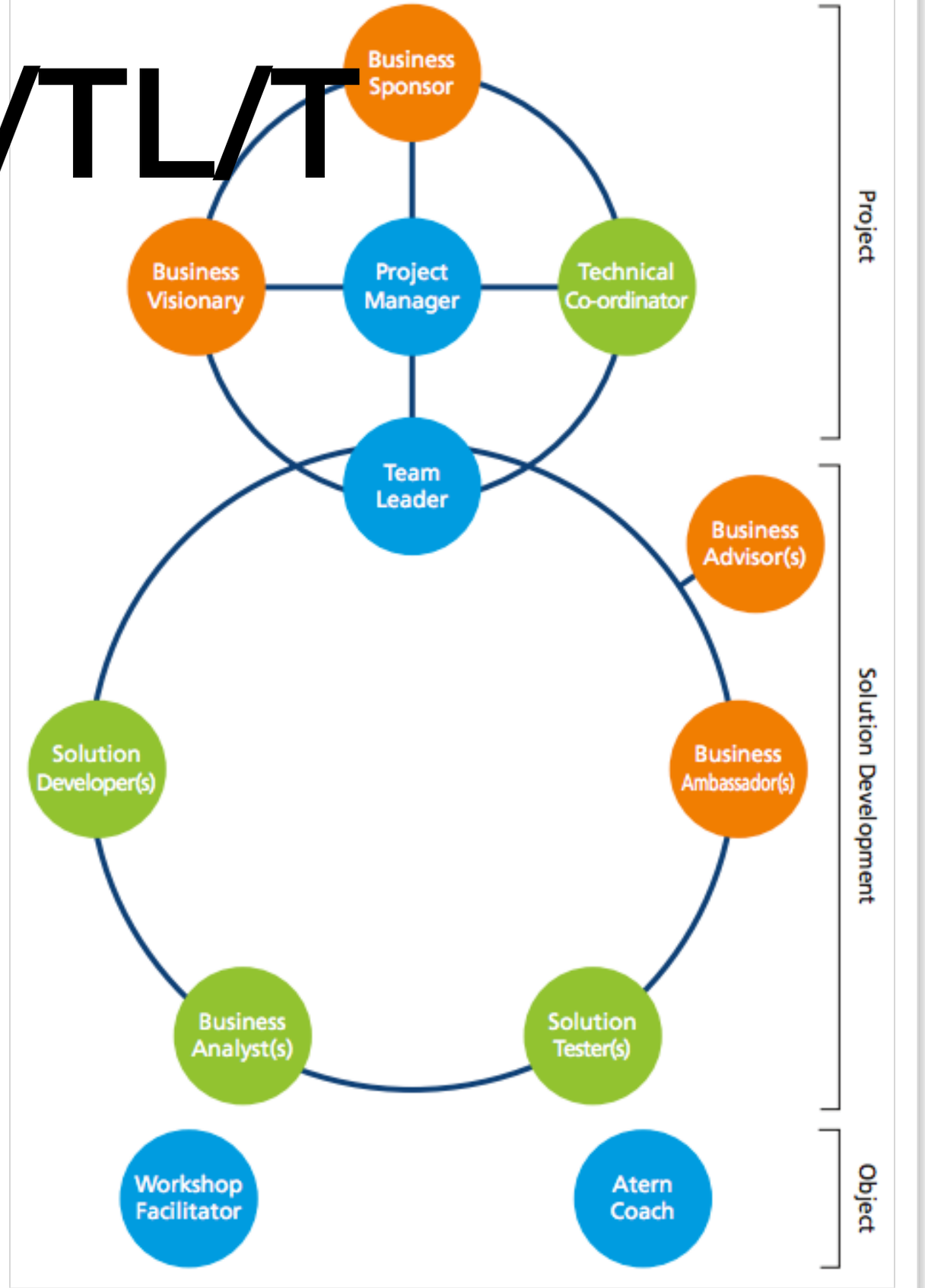


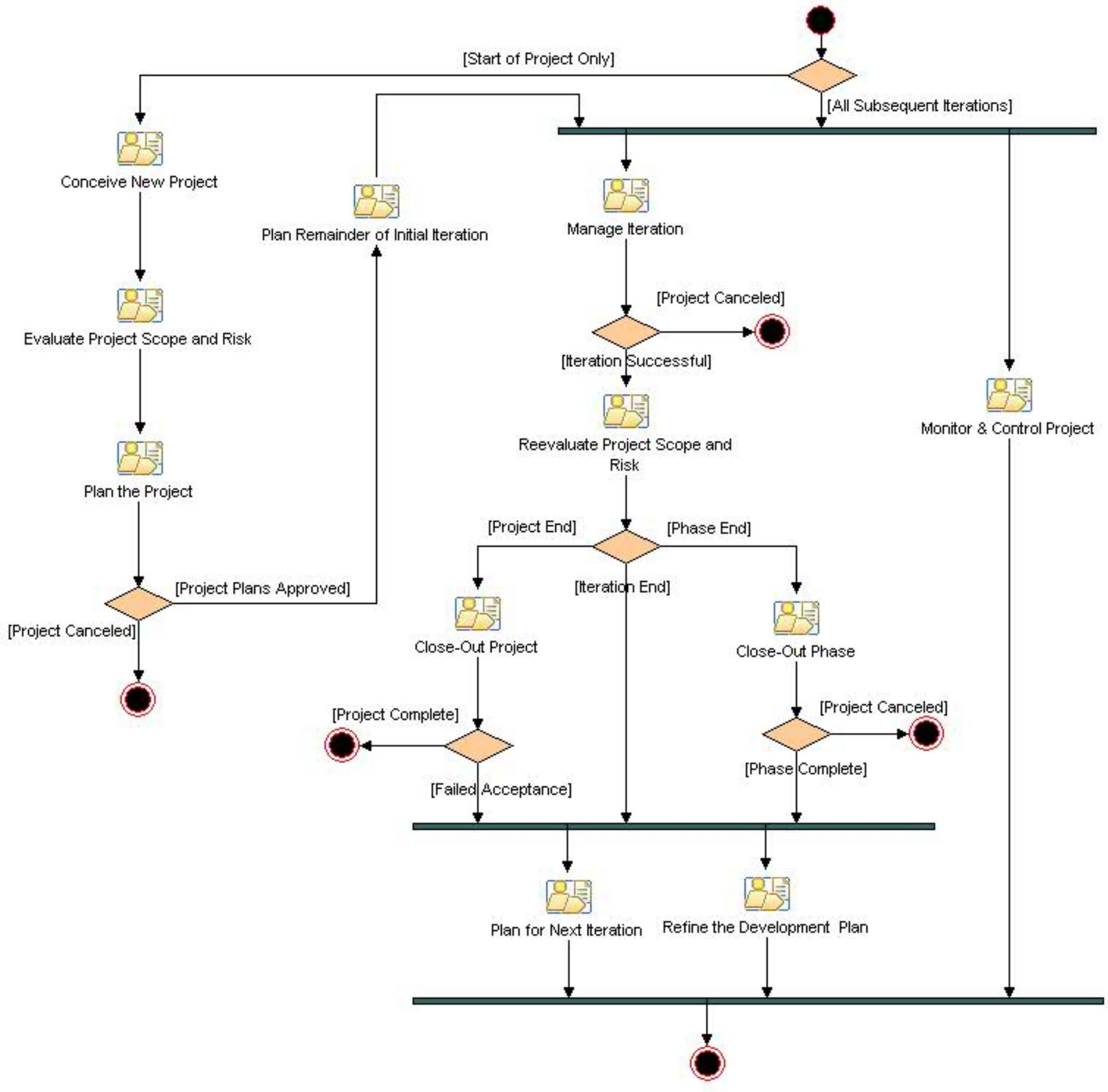
RUP a projektové řízení

Cíle:

- Poskytnout framework pro řízení softwarově orientovaných projektů.
- Poskytnout praktické průvodce pro plánování, obsazení pozic, vykonávání a monitorování projektu.
- Poskytnout nástroj pro řízení rizik.

PM/TL/T

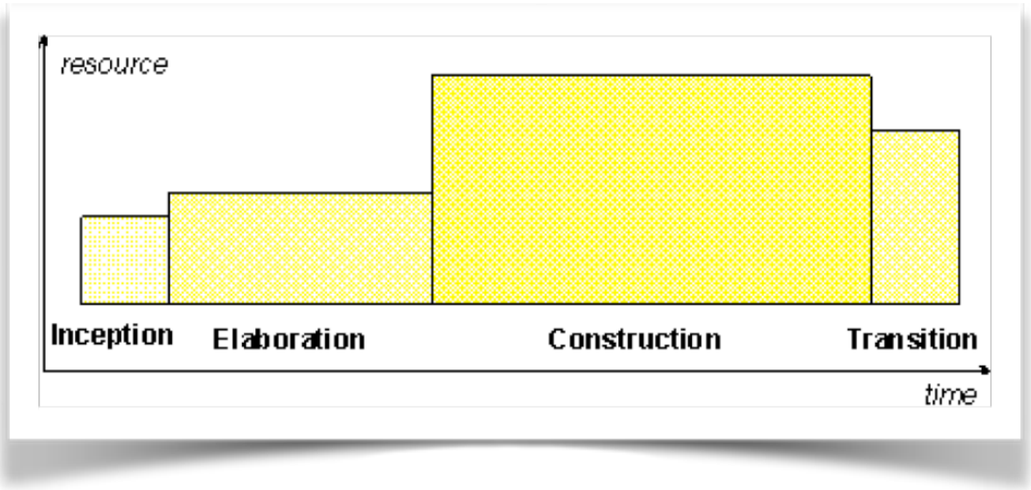

Business Sponsor
Business Visionary
Project Manager
Technical Co-ordinator
Team Leader
Business Advisor(s)
Solution Developer(s)
Business Ambassador(s)
Business Analyst(s)
Solution Tester(s)
Workshop Facilitator
Atern Coach
Project
Solution Development
Object


[Start of Project Only]
[All Subsequent Iterations]
Conceive New Project
Evaluate Project Scope and Risk
Plan the Project
[Project Plans Approved]
[Project Canceled]
Plan Remainder of Initial Iteration
Manage Iteration
[Project Canceled]
[Iteration Successful]
Reevaluate Project Scope and Risk
[Project End]
[Phase End]
[Iteration End]
Close-Out Project
Close-Out Phase
[Project Complete]
[Failed Acceptance]
[Project Canceled]
[Phase Complete]
Monitor & Control Project
Plan for Next Iteration
Refine the Development Plan

Kolik iterací, jak mají být dlouhé?

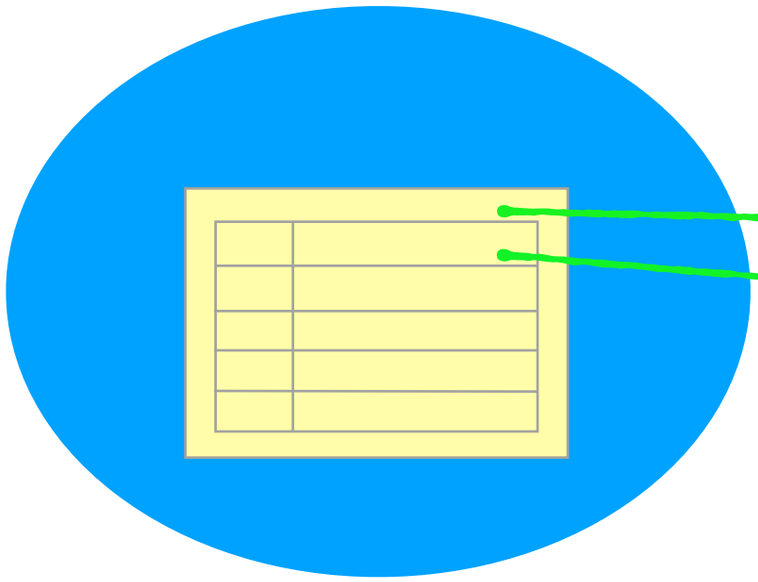
Záleží na:

- Doméně
- Zkušenostech týmu
- Nová či známá technologie?
- Známé



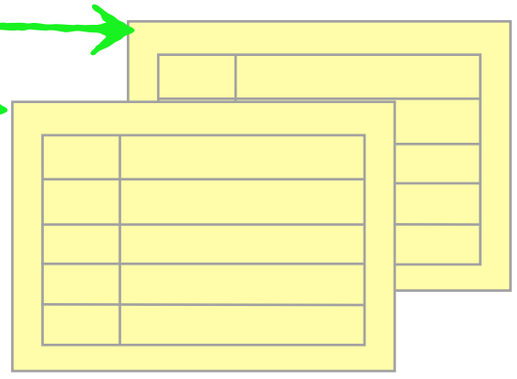
Řádek kódu	Počet lidí	Délka iterace
5 000	4	2 týdny
20 000	10	3-4 týdny
100 000	40 (sub-týmy)	4-6 týdnů
1 000 000	150 (sub-týmy)	4-6 týdnů

Dvě úrovně plánování



Projektový plán

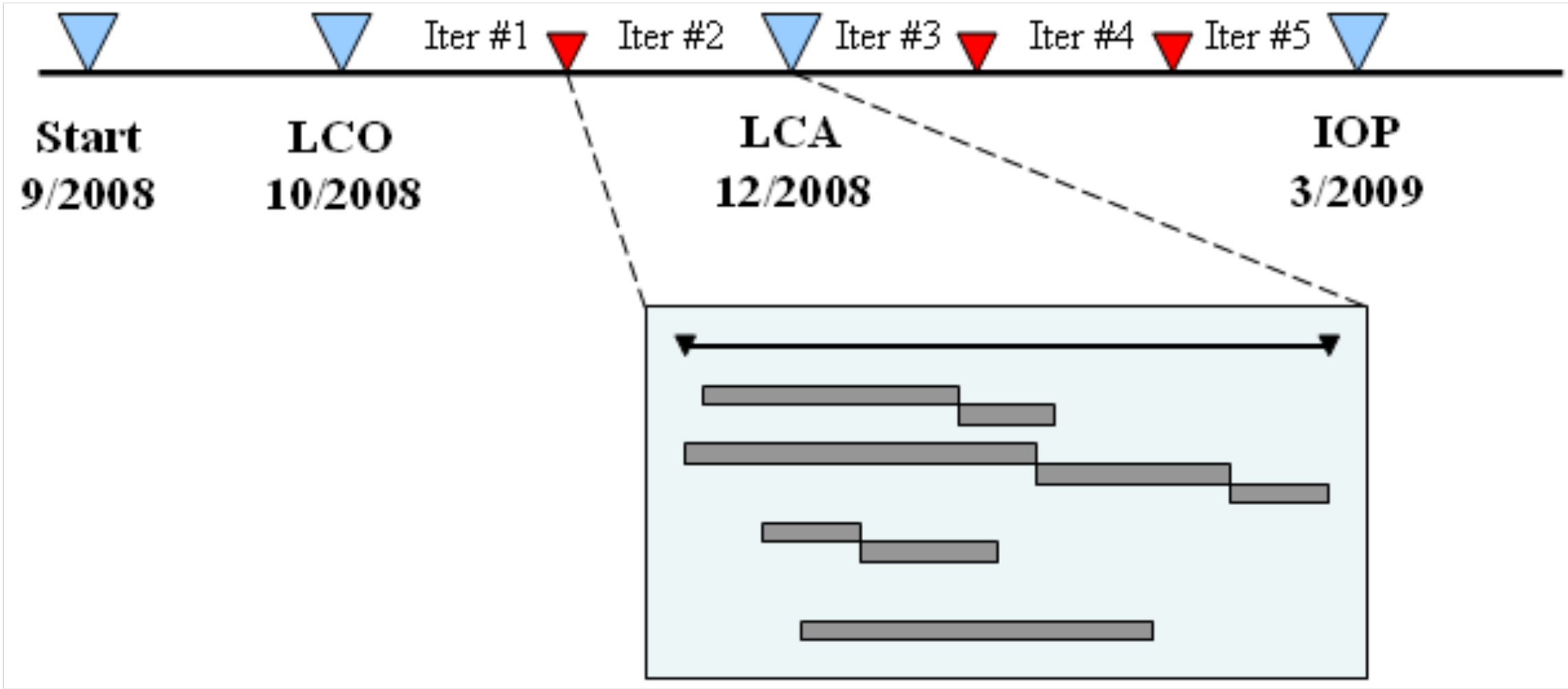
Iterační plán - současný



Iterační plán - budoucí

Pouze milníky a fáze (Road map)

Příklad


Iter #1
Iter #2
Iter #3
Iter #4
Iter #5
Start
9/2008
LCO
10/2008
LCA
12/2008
IOP
3/2009

Rizika a PM v RUP

- Vývoj SW založen hlavně na známých aspektech (tvorba plánu, definice a přiřazení úkolů).

- Neznámými aspekty se zabývá právě řízení rizik (tzv. Risk Management).

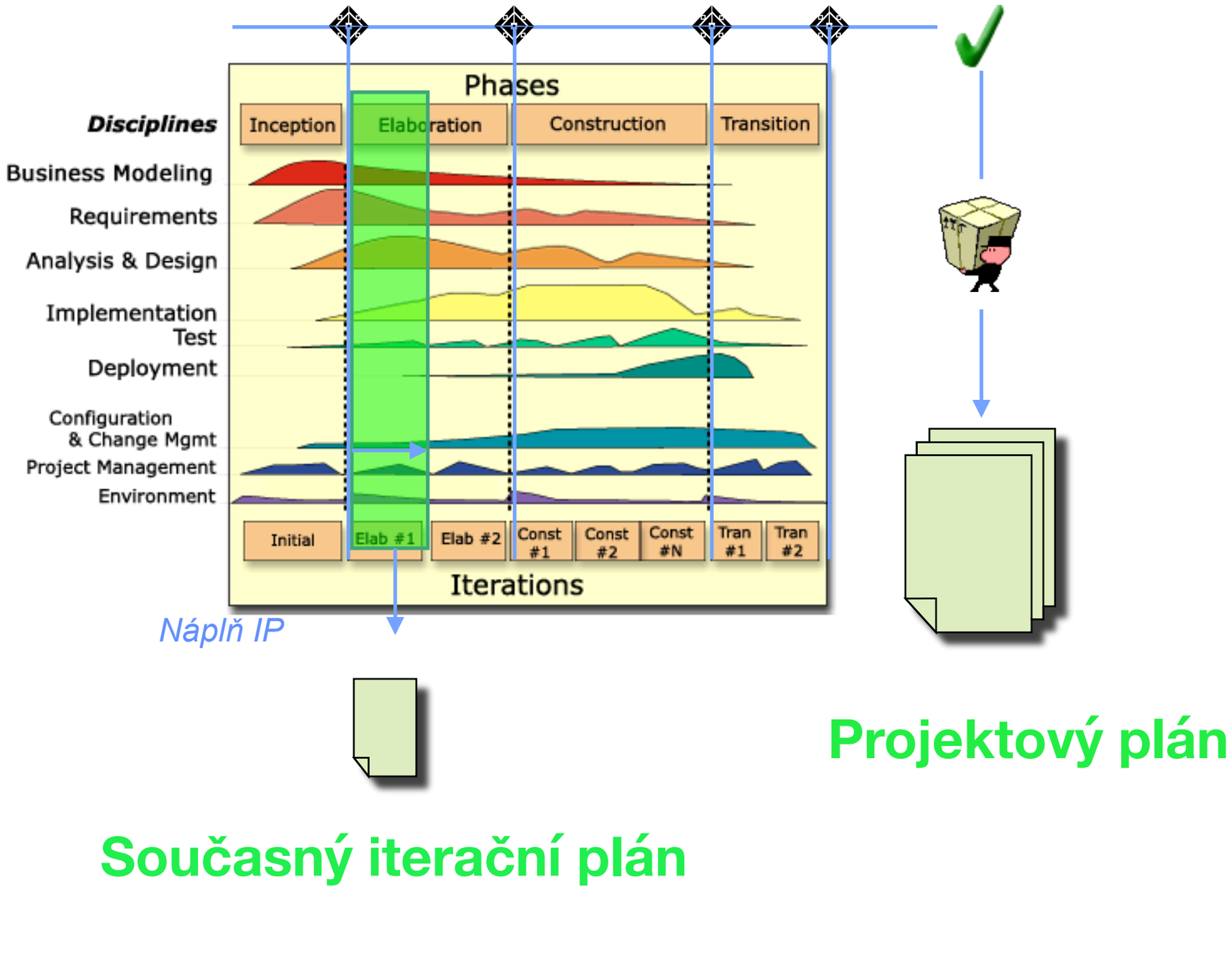
- Riziko – událost v softwarovém procesu, jejíž výskyt může zabránit úspěšnému doručení software v dohodnutém čase.

- Rizika jsou většinou nejistá, či dokonce neznámá, proto je nutné se jimi zabývat.

- Akce na snížení rizika

- Eventuální plán pro případ, že by riziko opravdu nastalo

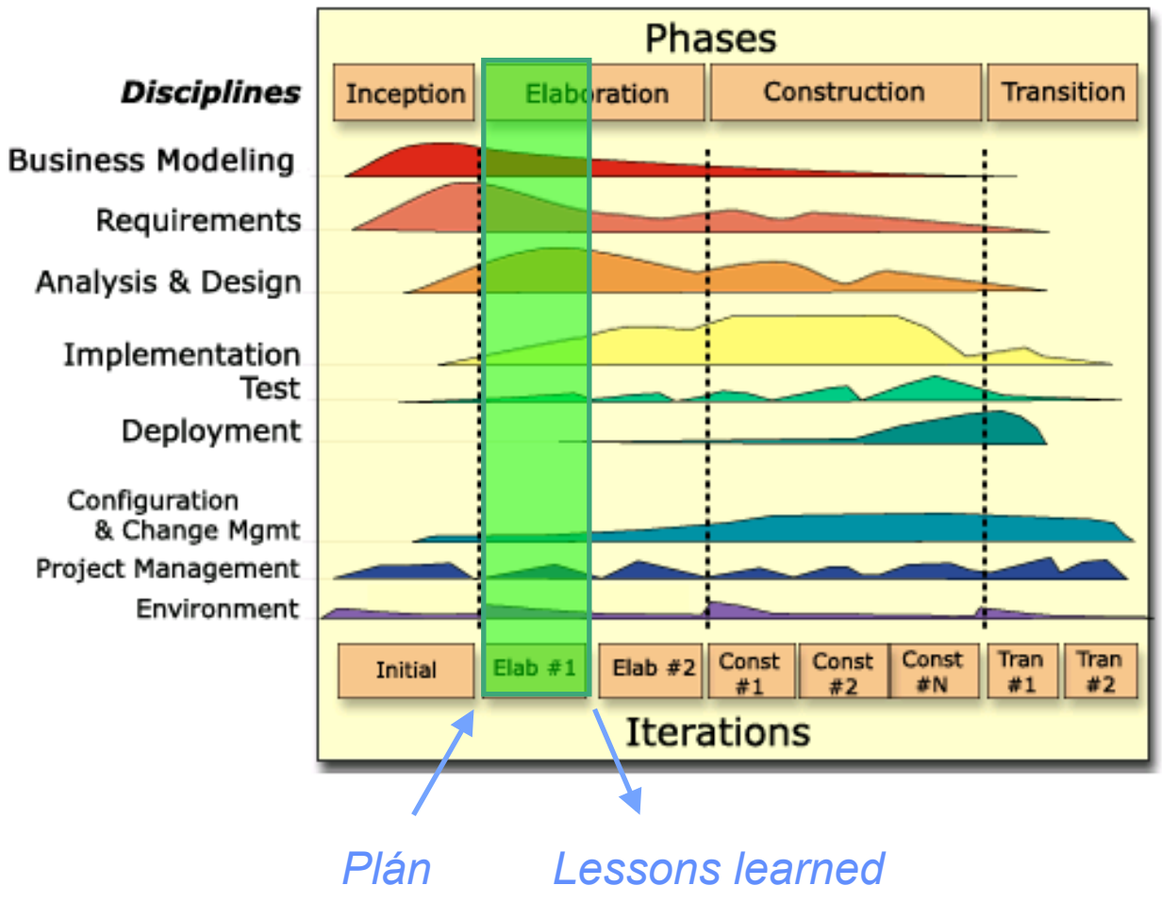
Projektový vs. iterační plán



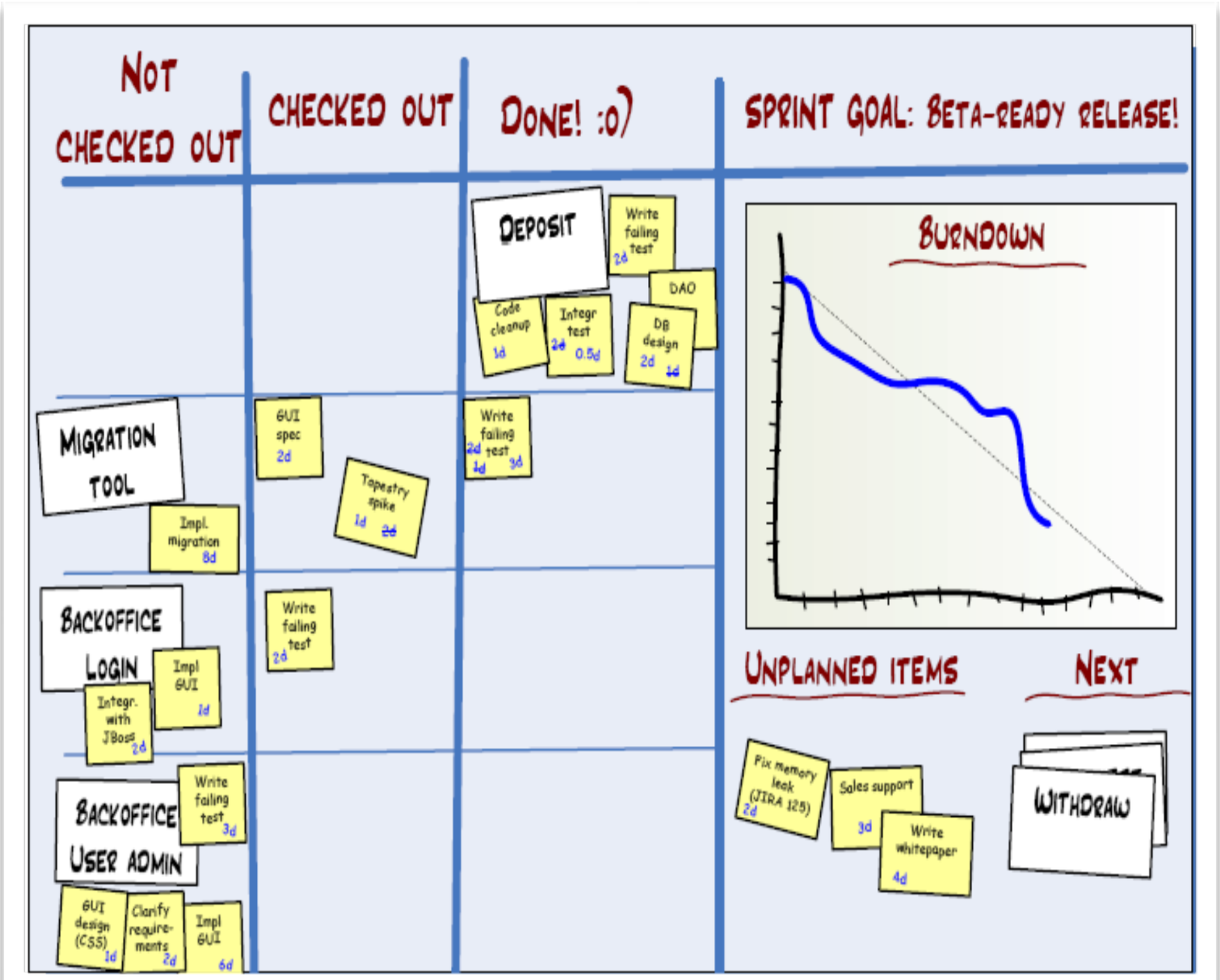
Současný iterační plán

Projektový plán

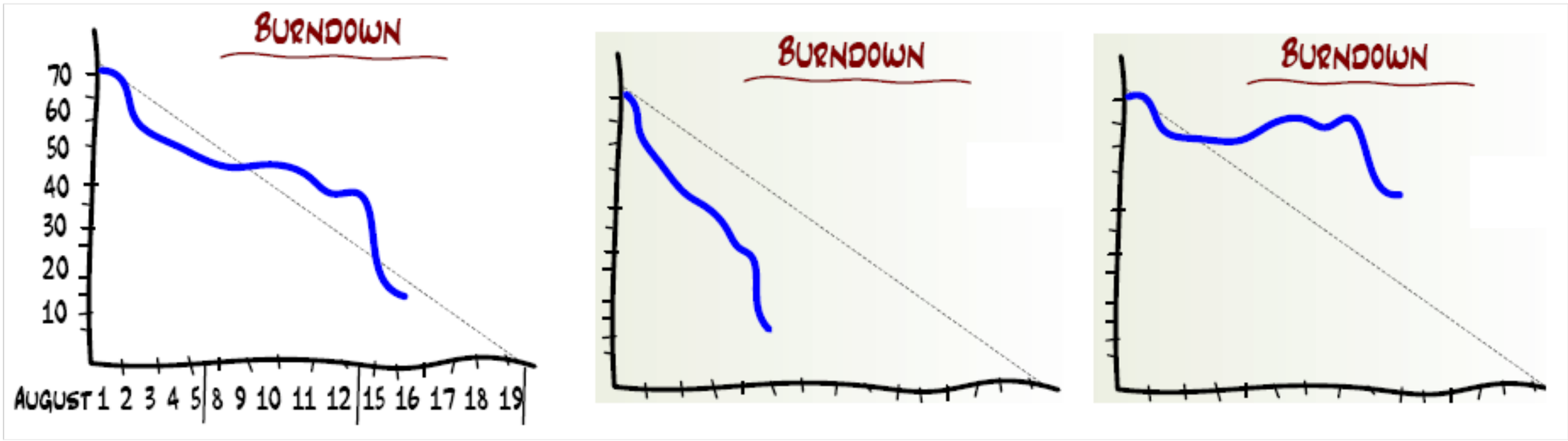
Timeboxed iterace



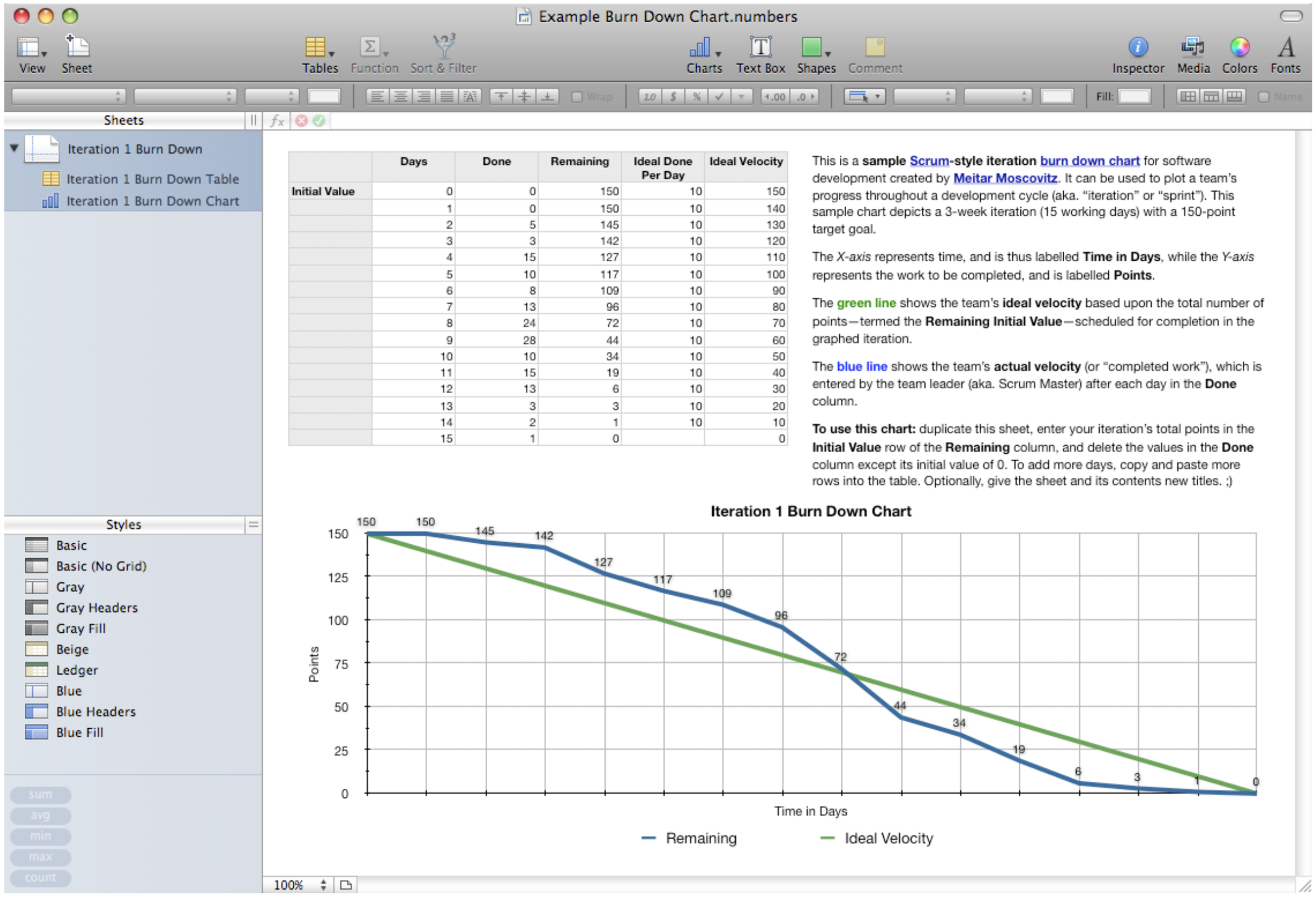
Burn-down chart



Identifikace problémů



Burn-down chart v praxi



	Days	Done	Remaining	Ideal Done Per Day	Ideal Velocity
Initial Value	0	0	150	10	150
	1	0	150	10	140
	2	5	145	10	130
	3	3	142	10	120
	4	15	127	10	110
	5	10	117	10	100
	6	8	109	10	90
	7	13	96	10	80
	8	24	72	10	70
	9	28	44	10	60
	10	10	34	10	50
	11	15	19	10	40
	12	13	6	10	30
	13	3	3	10	20
	14	2	1	10	10
	15	1	0		0

This is a **sample Scrum-style iteration burn down chart** for software development created by **Meitar Moscovitz**. It can be used to plot a team's progress throughout a development cycle (aka. "iteration" or "sprint"). This sample chart depicts a 3-week iteration (15 working days) with a 150-point target goal.

The *X-axis* represents time, and is thus labelled **Time in Days**, while the *Y-axis* represents the work to be completed, and is labelled **Points**.

The **green line** shows the team's **ideal velocity** based upon the total number of points—termed the **Remaining Initial Value**—scheduled for completion in the graphed iteration.

The **blue line** shows the team's **actual velocity** (or "completed work"), which is entered by the team leader (aka. Scrum Master) after each day in the **Done** column.

To use this chart: duplicate this sheet, enter your iteration's total points in the **Initial Value** row of the **Remaining** column, and delete the values in the **Done** column except its initial value of 0. To add more days, copy and paste more rows into the table. Optionally, give the sheet and its contents new titles. ;)

Retrospektiva

Základní otázky:

- Co fungovalo, co se osvědčilo v průběhu předchozí iterace (projektu, releasu)?

- Co moc dobře nefungovalo, co se neosvědčilo v průběhu předchozí iterace (projektu, releasu)?

- Co bychom měli dělat jinak, jaké akce by [illegible] podniknout za účelem zlepšení v další it[illegible] releasu)?

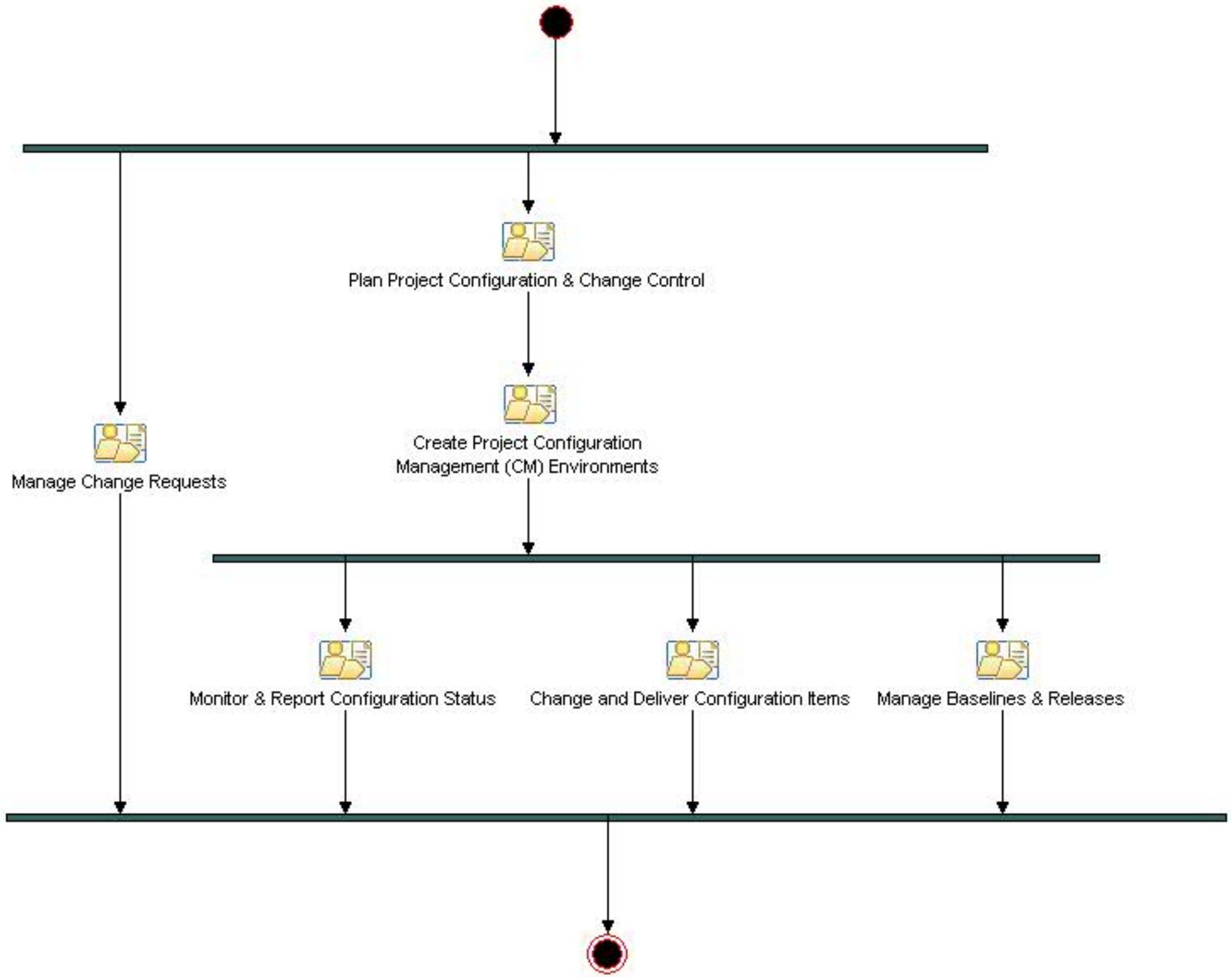
Retrospektiva

Configuration & Change management

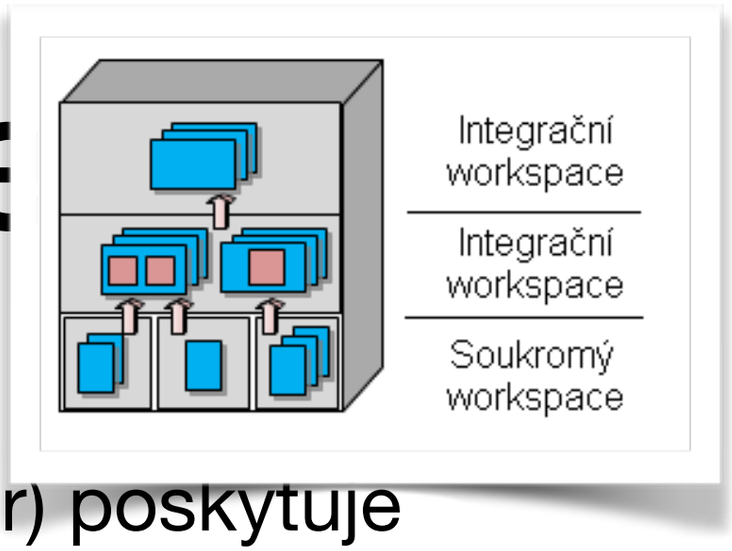
CCM

Hlavní aktivity:

- Plánování konfigurace a řízení změn
  - Vytvoření politik pro CM a CxM
  - Dokumentování těchto informací
- Vytvoření prostředí pro správu konfigurací
  - Zajištění nezbytných artefaktů pro vývojáře a integrátory (soukromé a veřejné workspace, repository a její struktura, HW a SW pro buildování)
  - Těsná kooperace se softwarovým architektem


Plan Project Configuration & Change Control
Manage Change Requests
Create Project Configuration Management (CM) Environments
Monitor & Report Configuration Status
Change and Deliver Configuration Items
Manage Baselines & Releases

Workspace



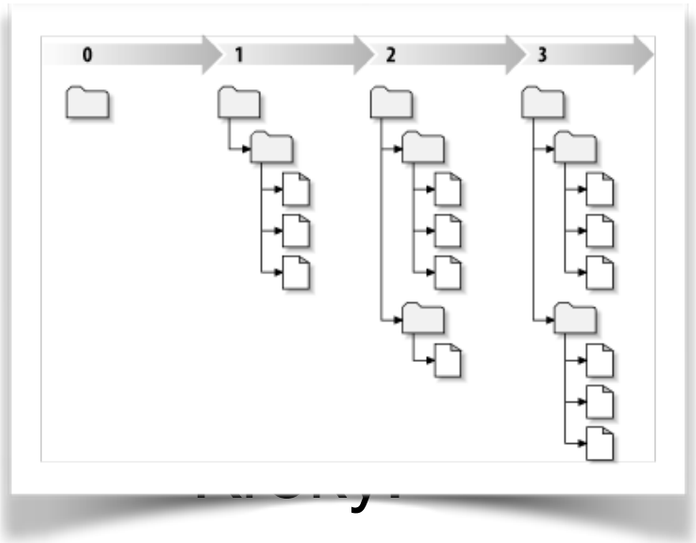
- Workspace (česky privátní pracovní prostor) poskytuje jednotlivým vývojářům či malým týmům prostředí, ve kterém mohou pracovat jako by pracovali izolovaně, odděleně.
- V rámci tohoto prostoru přístup ke všem potřebným artefaktům.
- Workspace slouží pro základní vývoj malých dílků a pro jejich postupnou integraci.
- Výsledkem integrace několika workspace je další inkrement aplikace.

Nástroje

V rámci této disciplíny:

- Repository
  - pro uložení artefaktů
  - umožnění paralelní práce programátorů (CVS, SVN, IBM Rational ClearCase)
  - možnost napojení na build mechanismus (Ant, Maven, ClearCase).

- Podpora změnového řízení
  - evidence změn a defektů (issue tracking tool) jako jsou Jira, IBM Rational ClearQuest.

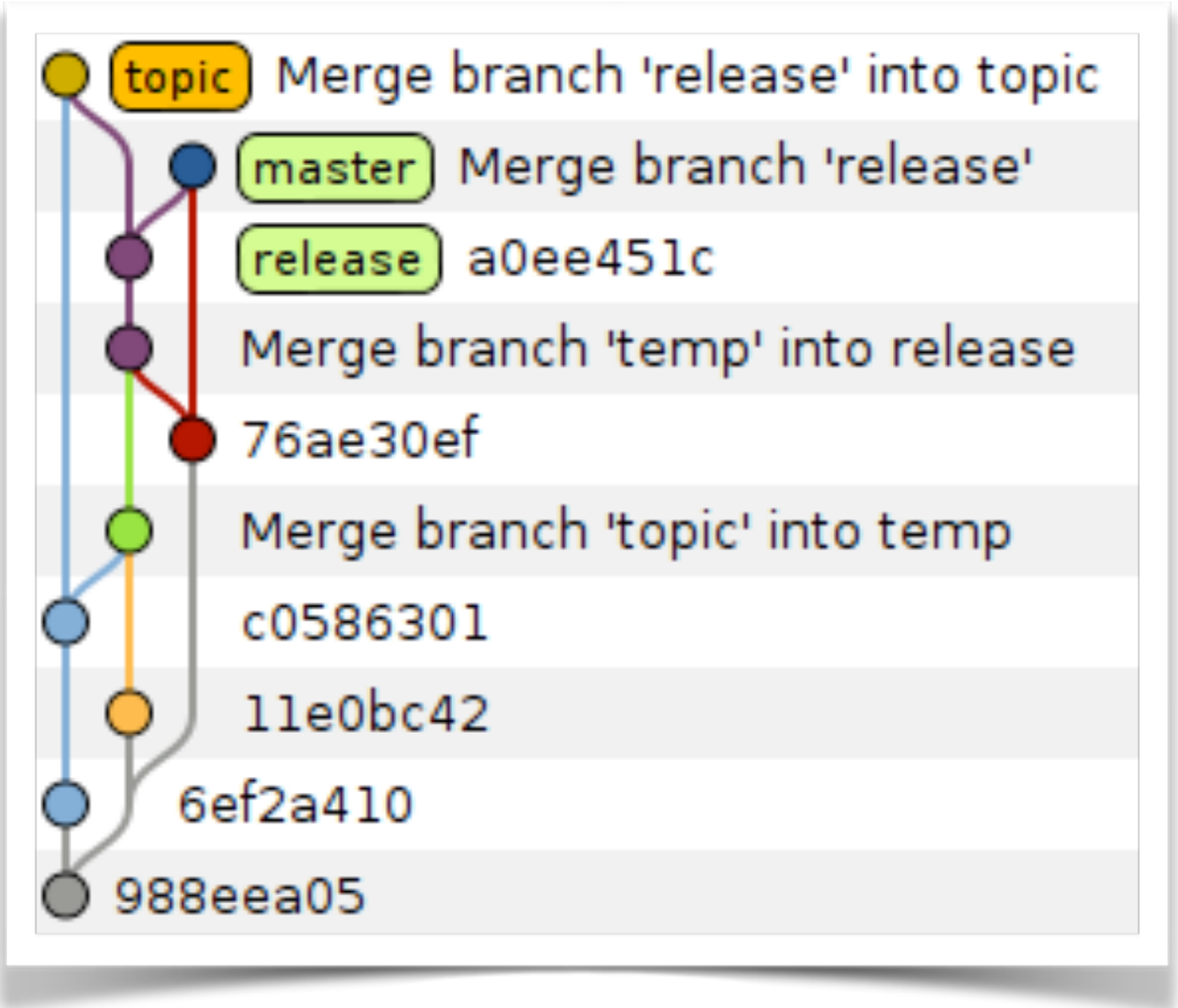


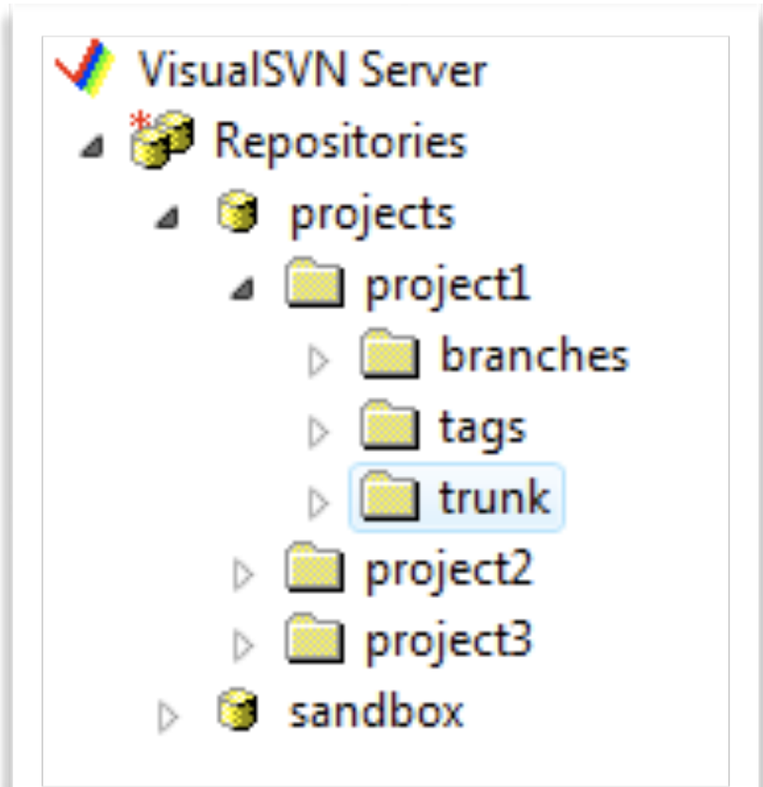
Repository

Každá potvrzená změna vytvoří novou verzi projektu.
(pozor na větve)

1. získání pracovní kopie z repozitory (svn checkout)
2. modifikace pracovní kopie
3. aktualizace - získání změn v repozitory (svn update)
4. další modifikace pracovní kopie
5. zjištění rozdílu pracovní kopie a repozitory (svn diff)
6. publikace změn v pracovní kopii do repozitory (svn commit)

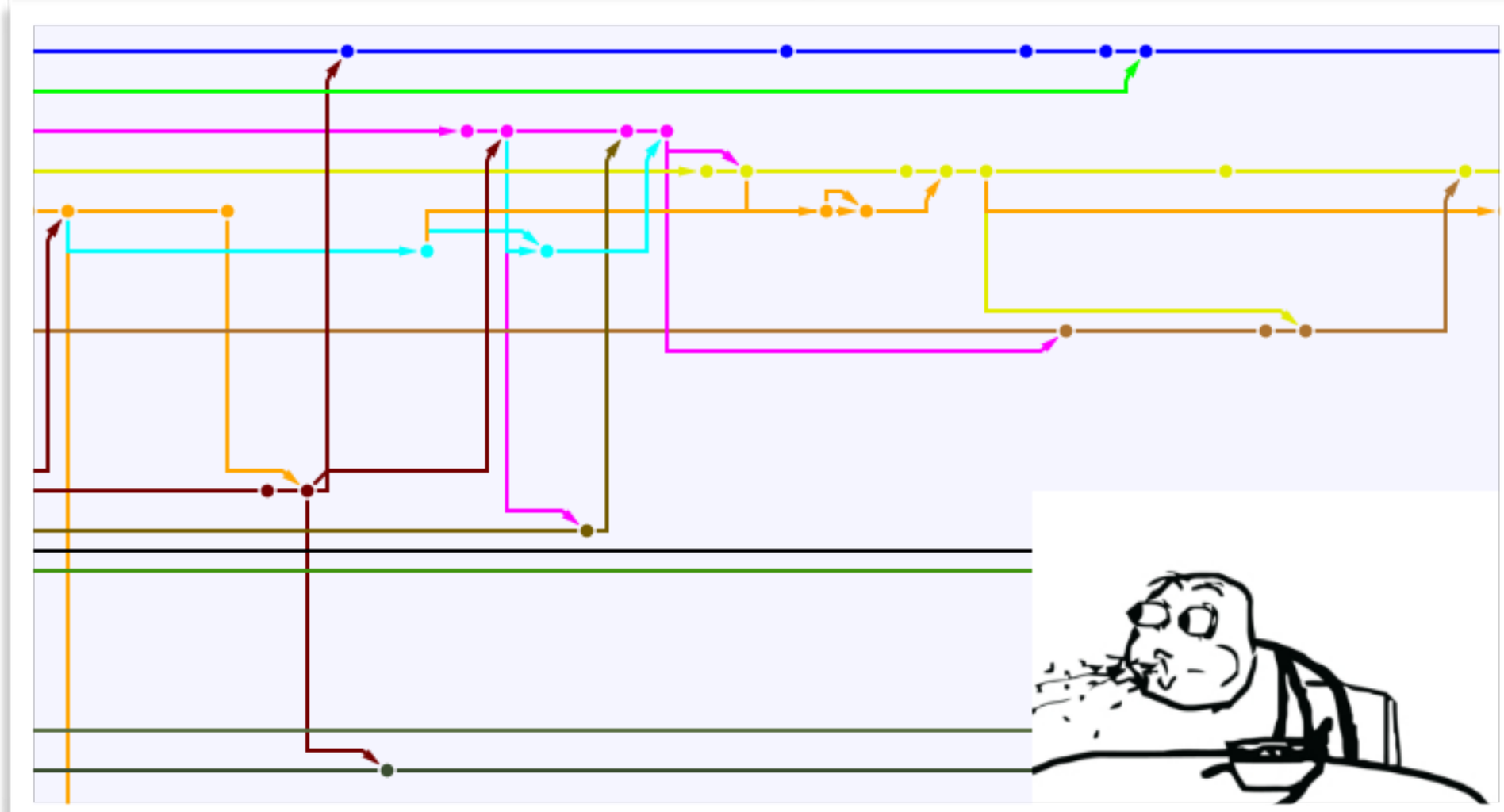
Struktura repository

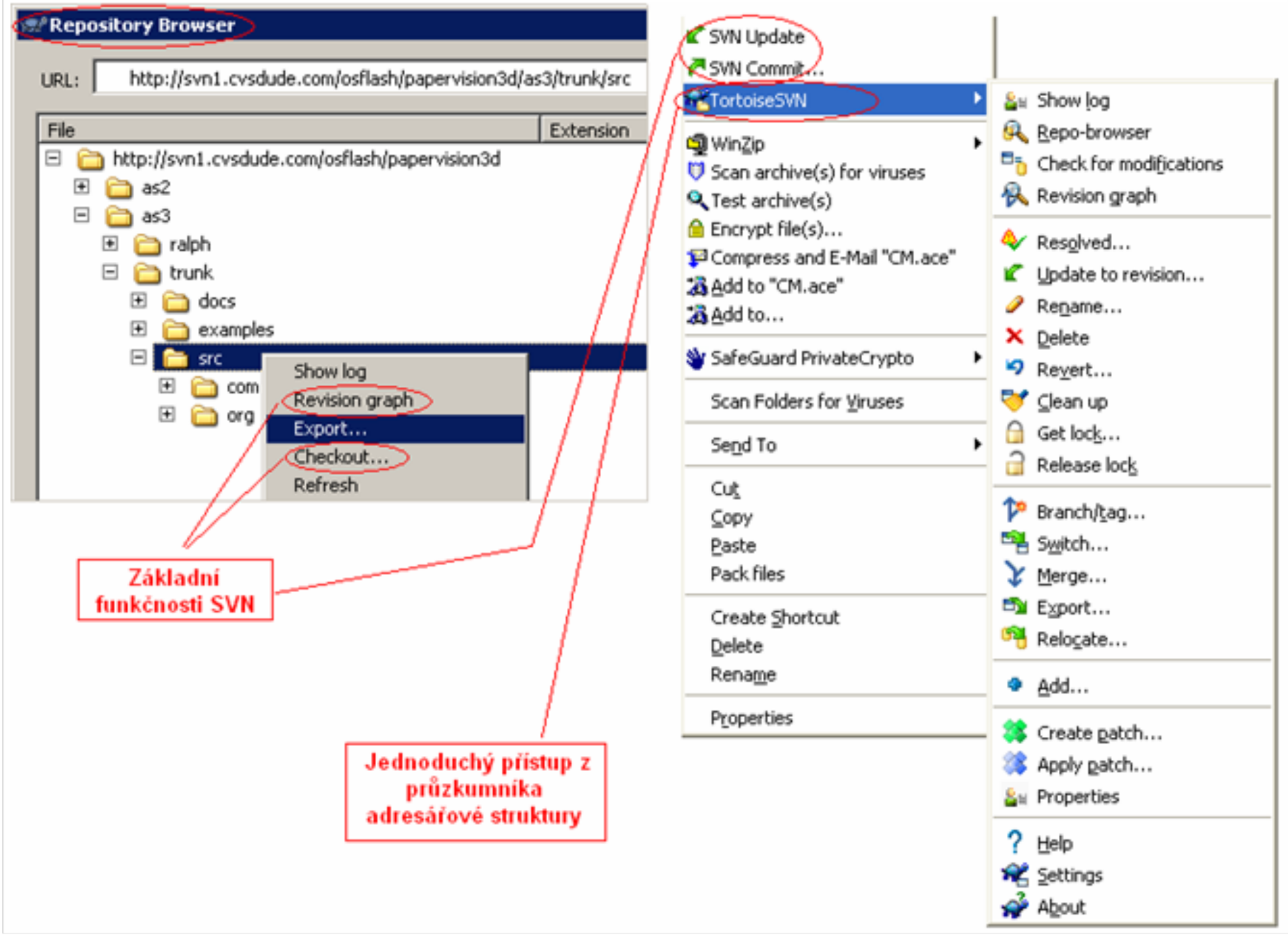




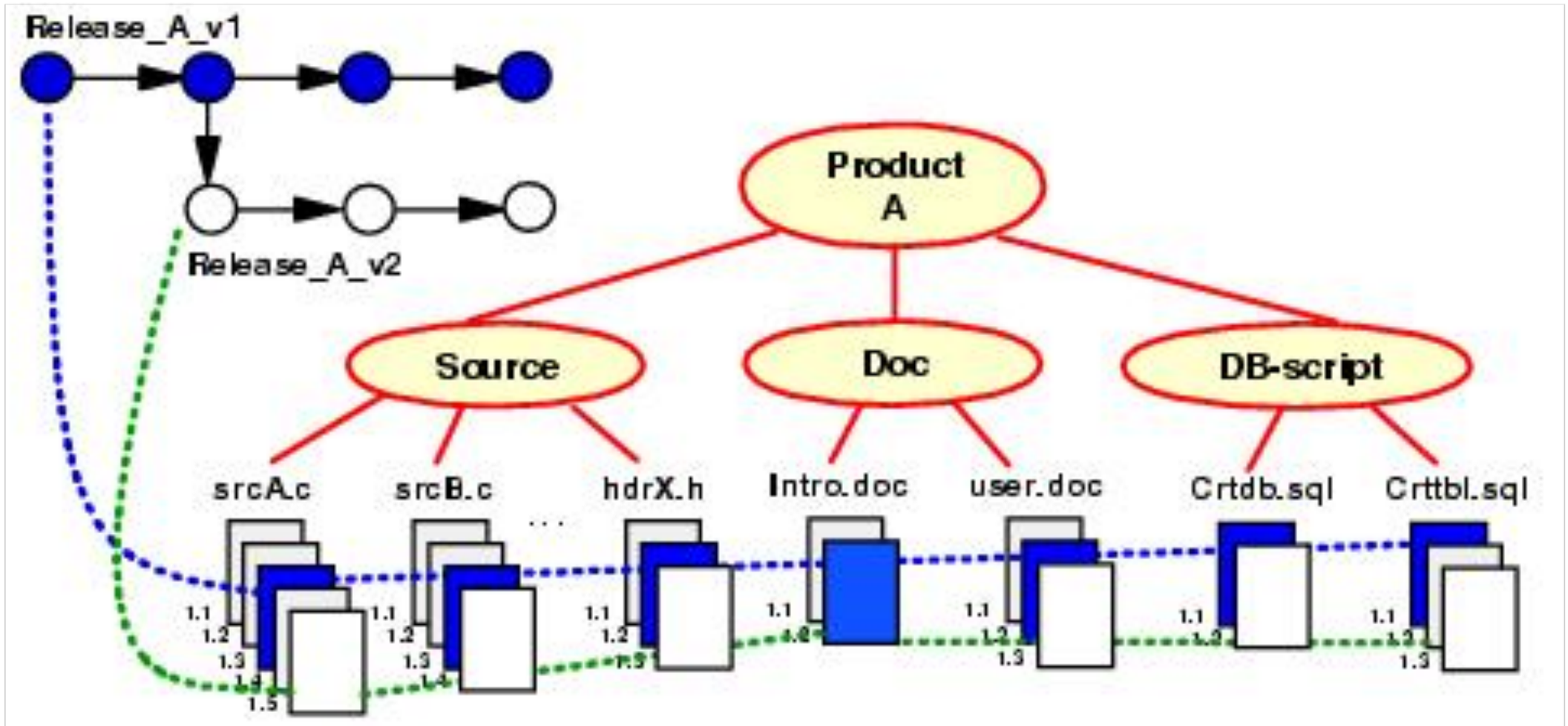
Git flow

Merge hell




Repository Browser
URL:
http://svn1.cvsdude.com/osflash/papervision3d/as3/trunk/src
File
Extension
http://svn1.cvsdude.com/osflash/papervision3d
as2
as3
ralph
trunk
docs
examples
src
com
org
Show log
Revision graph
Export...
Checkout...
Refresh
SVN Update
SVN Commit...
TortoiseSVN
WinZip
Scan archive(s) for viruses
Test archive(s)
Encrypt file(s)...
Compress and E-Mail "CM.ace"
Add to "CM.ace"
Add to...
SafeGuard PrivateCrypto
Scan Folders for Viruses
Send To
Cut
Copy
Paste
Pack files
Create Shortcut
Delete
Rename
Properties
Show log
Repo-browser
Check for modifications
Revision graph
Resolved...
Update to revision...
Rename...
Delete
Revert...
Clean up
Get lock...
Release lock
Branch/tag...
Switch...
Merge...
Export...
Relocate...
Add...
Create patch...
Apply patch...
Properties
Help
Settings
About
Základní funkčnosti SVN
Jednoduchý přístup z průzkumníka adresářové struktury

Release z repository



AR AccuRev

File Edit Actions View Issues Tools Admin Help

Open Workspace | View Streams | View Issue Queries | Create Issue | Open Issue | Refresh View | Help

Welcome | [redacted] | SRC | [redacted]

Mode: Workspace Explorer | Outgoing | Incoming | Conflicts | Save Layout

Update | Open Parent [redacted] | Show Diff Pane

Incoming Changes Mode

- .
  - Websites
  - Styling

Filter Table

In Folder	Name	Status	Real	Action	Id	New Name
\.\Website[redacted]rap.Site\Scripts...	Resources		29617/1	create	1047849	
\.\Website[redacted]rap.Site\Scripts...	JuniorIsa		29617/1	create	1047845	
\.\Website[redacted]rap.Site\Scripts...	ImportantInformation		29617/1	create	1047843	
\.\Website[redacted]rap.Site\Scripts...	Index.tsx		29434/2	replace content	1047717	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	Index.tsx		29434/2	replace content	1047715	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	SellAllocation.tsx		29434/3	replace content	1043304	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	Index.tsx		29434/15	replace content	1027769	\.\Website[redacted]
\.\Website[redacted]rap.Site\Areas\...	AccountDocumentUploadCo...		28027/1	replace content	1025487	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	Content.ts		28027/1	replace content	1025180	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	Content.ts		28027/1	replace content	1023786	\.\Website[redacted]
\.\Website[redacted]rap.Site\Scripts...	MarketLightbox.tsx		28027/1	replace content	1024611	\.\Website[redacted]
\.\Styling[redacted]	_components.table.investme...		23975/3	replace content	1017714	\.\Styling[redacted]
\.\Website[redacted]Site\Scripts...	Investments.tsx		30375/1	replace content	991947	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	PersonalPerformance.tsx		30375/2	replace content	991701	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	SideNavigationMenu.tsx		30530/1	replace content	988191	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	AvailableCash.tsx		30375/1	replace content	991771	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	Content.ts		28027/1	replace content	966681	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	InstructionTypeDictionary.ts		28027/1	replace content	1026782	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	MarketSettings.tsx		28027/1	replace content	1023767	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	Index.tsx		29434/2	replace content	1027594	\.\Website[redacted]
\.\Website[redacted]Site\Scripts...	Transfers.tsx		30375/1	replace content	993074	\.\Website[redacted]
\.\Website[redacted]Scripts...	Index.tsx		29434/17	replace content	1027100	\.\Website[redacted]
\.\Website[redacted]Scripts...	Index.tsx		30530/1	replace content	1014727	\.\Website[redacted]
\.\Website[redacted]Areas\...	StartProcessController.cs		29380/1	replace content	996411	\.\Website[redacted]
\.\Website[redacted]	Fnz.Wrap.Site.csproj		29617/11	replace content	821288	\.\Website[redacted]
\.\Website[redacted]Scripts...	Index.tsx		28027/1	replace content	995653	\.\Website[redacted]
\.\Website[redacted]Scripts...	Transactions.tsx		30375/1	replace content	992979	\.\Website[redacted]
\.\Website[redacted]Scripts...	CharacterCounter.tsx		28027/1	replace content	996501	\.\Website[redacted]
\.\Website[redacted]Scripts...	StartLink.tsx		29617/1	create	1047858	
\.\Website[redacted]Scripts...	Residency.tsx		29617/1	create	1047857	
\.\Website[redacted]Scripts...	JuniorResidency.tsx		29617/1	create	1047853	
\.\Website[redacted]Scripts...	Content.ts		29617/1	create	1047851	
\.\Website[redacted]Scripts...	Validation.ts		29617/1	create	1047850	
\.\Website[redacted]Scripts...	DateOfBirth.tsx		29617/1	create	1047848	
\.\Website[redacted]Scripts...	Variables.ts		29617/1	create	1047846	
\.\Website[redacted]Scripts...	Information.tsx		29617/1	create	1047844	
\.\Website[redacted]Scripts...	Documents.tsx		29617/1	create	1047859	
\.\Website[redacted]Scripts...	Urls.ts		29617/1	create	1047852	
\.\Website[redacted]Scripts...	HasChildTrust.tsx		29617/1	create	1047855	
\.\Website[redacted]Scripts...	HasChildTrustConfirmation.tsx		29617/1	create	1047854	
\.\Website[redacted]Scripts...	StartProcessActions.ts		29617/1	create	1047856	
\.\Website[redacted]Scripts...	Index.tsx		29617/1	create	1047847	
\.\Website[redacted]Scripts...	JuniorIsa		29617/1	create	1047842	

Total items: 43

[redacted] | SRC | [redacted]

Change management

...proces pro registraci, ohodnocení, schválení či zamítnutí a implementování změn;

požadavek na změnu (RfC – Request for Change) - popsaný návrh změny jednoho nebo více artefaktů;

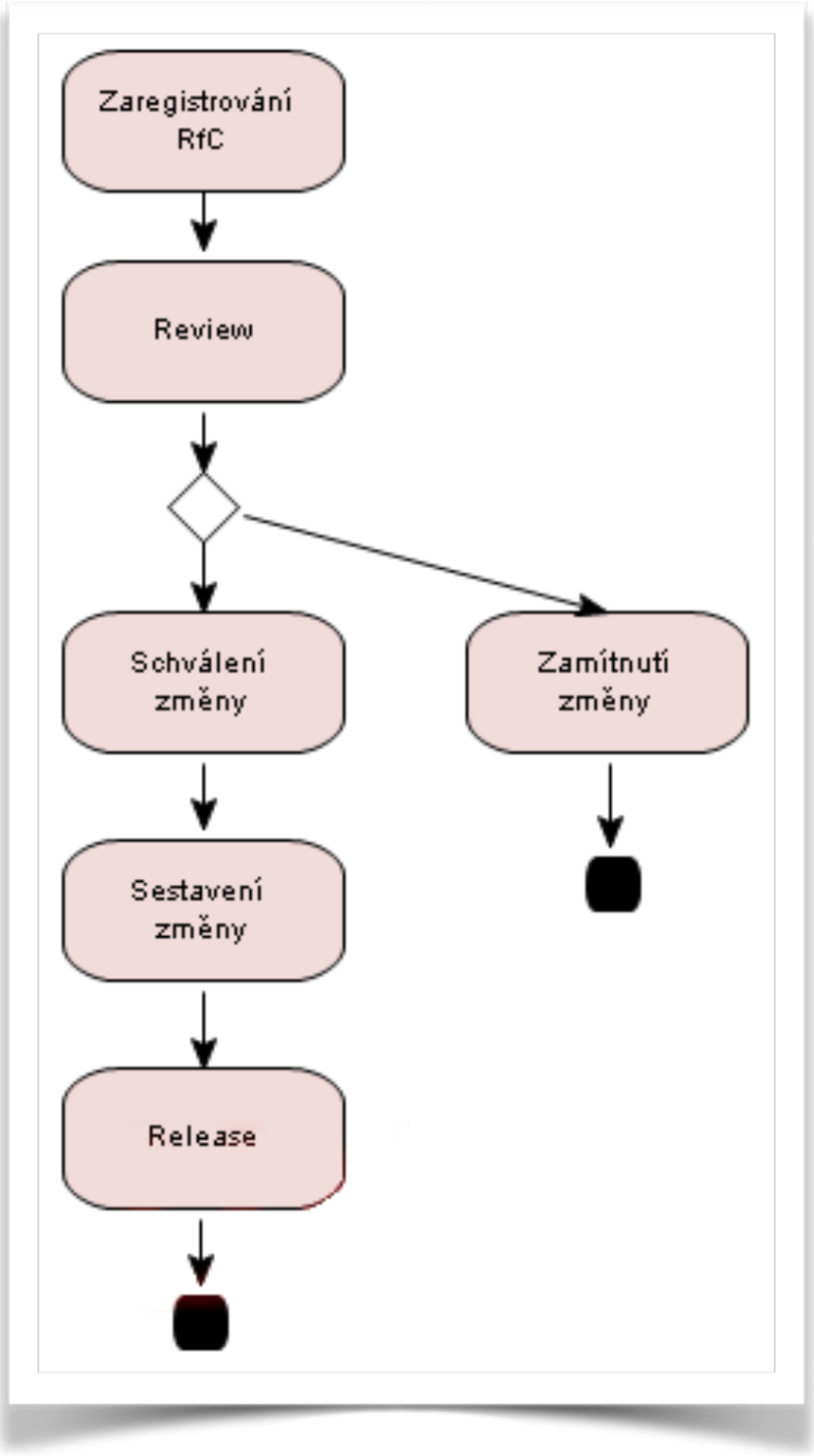
RfC vyvoláno z mnoha důvodů:

- oprava chyby,
- zlepšení kvality produktu (výkonnost, použitelnost),
- přidání nového požadavku.

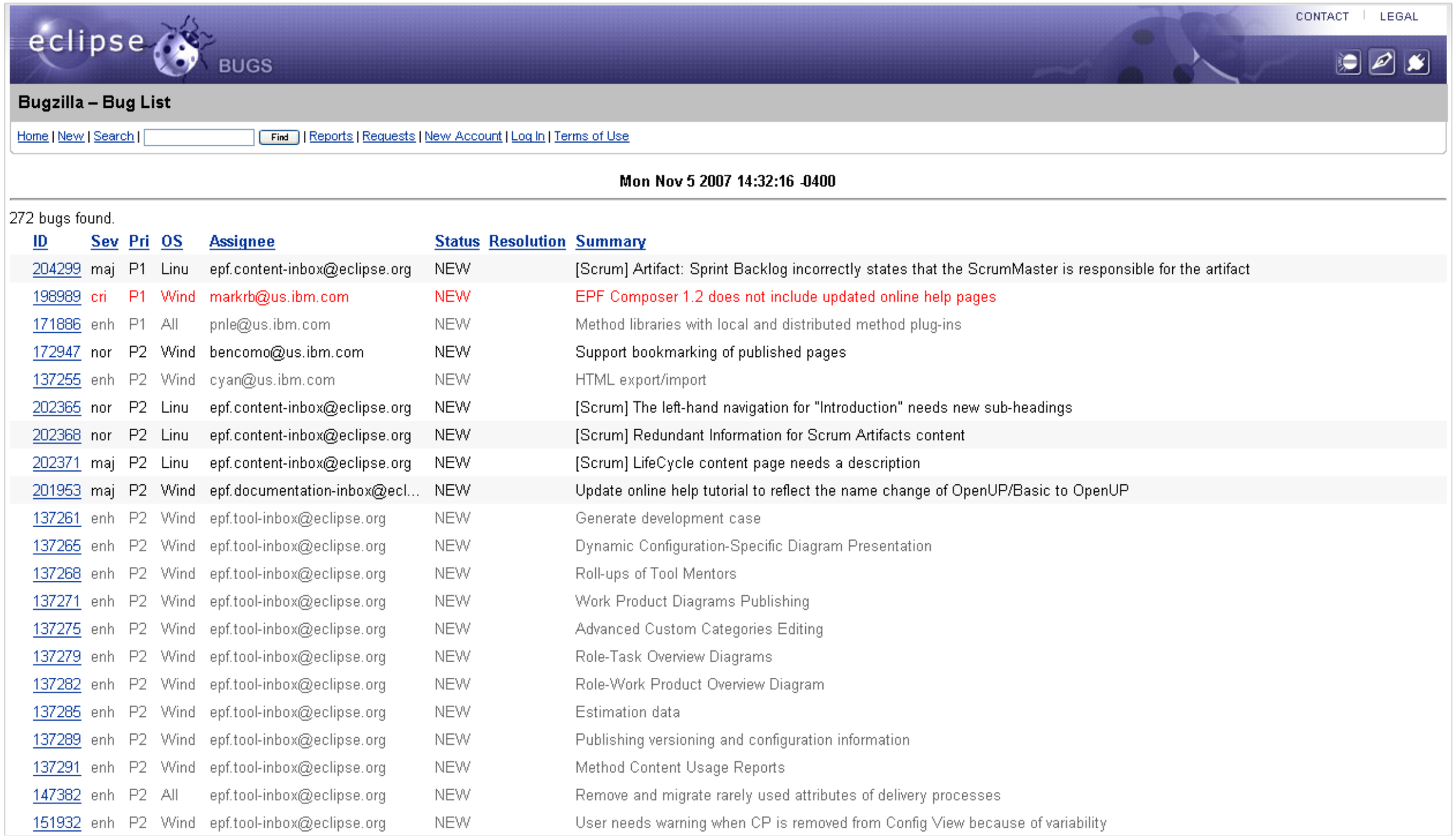
Požadavek a jeho životní cyklus

- nový -> zaevidovaný -> schválený -> přiřazený -> zpracovávaný -> hotový
- při změně stavu doplňovány informace (důvod, dopad změny, dopad na návrh/architekturu, odhad ceny změny)

Workflow

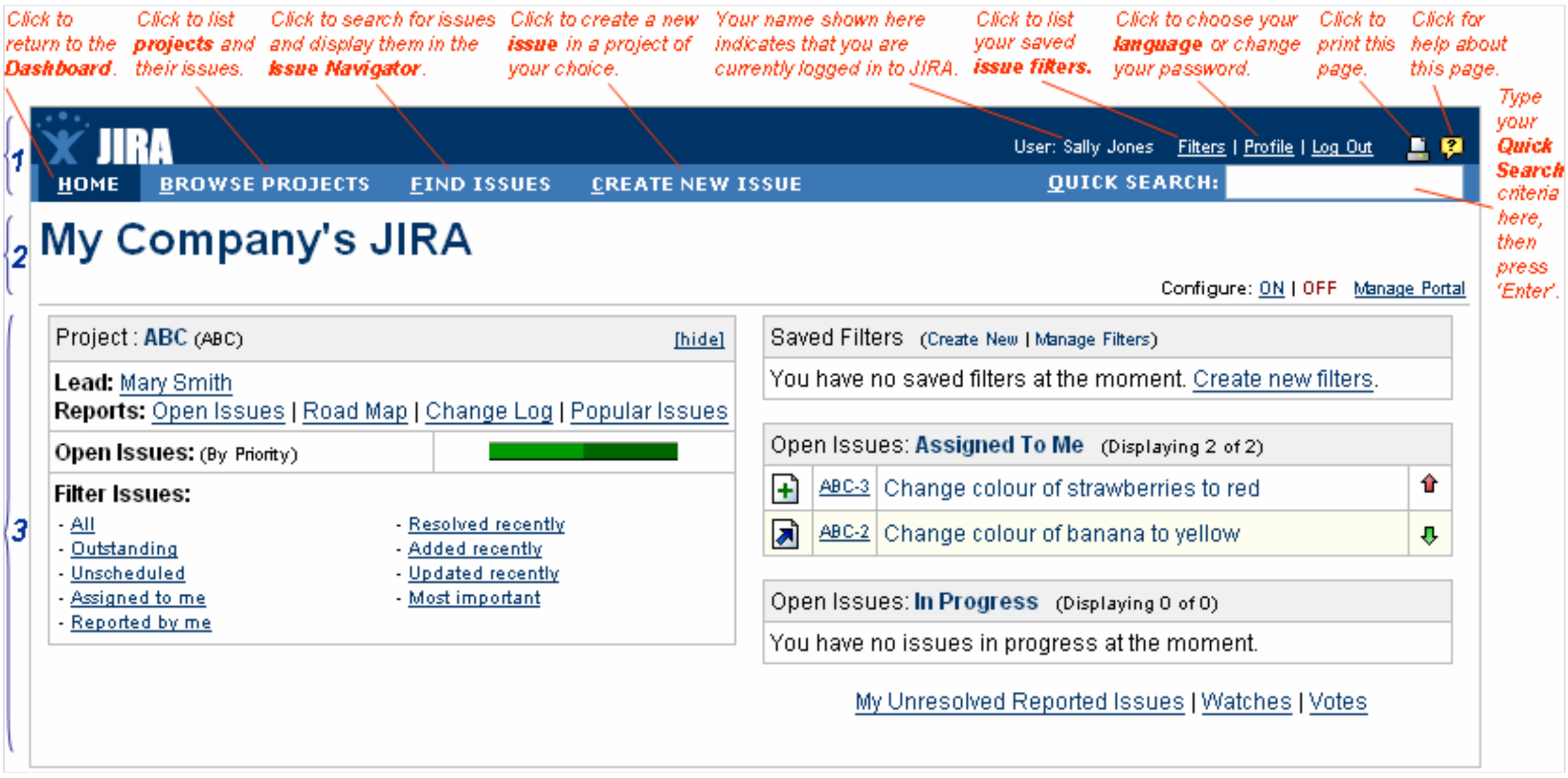

Zaregistrování RfC
Review
Schválení změny
Zamítnutí změny
Sestavení změny
Release

Bugs, change request



ID	Sev	Pri	OS	Assignee	Status	Resolution	Summary
204299	maj	P1	Linu	epf.content-inbox@eclipse.org	NEW		[Scrum] Artifact: Sprint Backlog incorrectly states that the ScrumMaster is responsible for the artifact
198989	cri	P1	Wind	markrb@us.ibm.com	NEW		EPF Composer 1.2 does not include updated online help pages
171886	enh	P1	All	pnle@us.ibm.com	NEW		Method libraries with local and distributed method plug-ins
172947	nor	P2	Wind	bencomo@us.ibm.com	NEW		Support bookmarking of published pages
137255	enh	P2	Wind	cyan@us.ibm.com	NEW		HTML export/import
202365	nor	P2	Linu	epf.content-inbox@eclipse.org	NEW		[Scrum] The left-hand navigation for "Introduction" needs new sub-headings
202368	nor	P2	Linu	epf.content-inbox@eclipse.org	NEW		[Scrum] Redundant Information for Scrum Artifacts content
202371	maj	P2	Linu	epf.content-inbox@eclipse.org	NEW		[Scrum] LifeCycle content page needs a description
201953	maj	P2	Wind	epf.documentation-inbox@ecl...	NEW		Update online help tutorial to reflect the name change of OpenUP/Basic to OpenUP
137261	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Generate development case
137265	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Dynamic Configuration-Specific Diagram Presentation
137268	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Roll-ups of Tool Mentors
137271	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Work Product Diagrams Publishing
137275	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Advanced Custom Categories Editing
137279	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Role-Task Overview Diagrams
137282	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Role-Work Product Overview Diagram
137285	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Estimation data
137289	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Publishing versioning and configuration information
137291	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		Method Content Usage Reports
147382	enh	P2	All	epf.tool-inbox@eclipse.org	NEW		Remove and migrate rarely used attributes of delivery processes
151932	enh	P2	Wind	epf.tool-inbox@eclipse.org	NEW		User needs warning when CP is removed from Config View because of variability

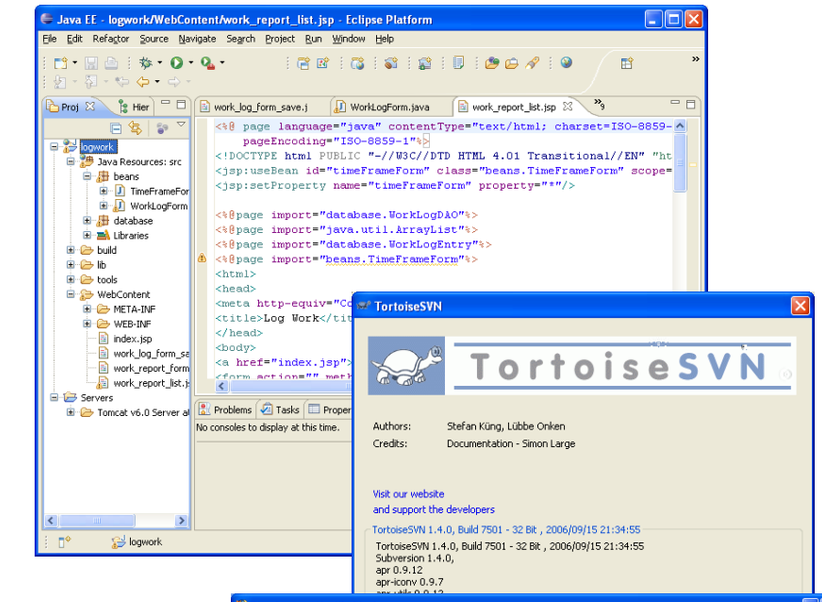
Jira

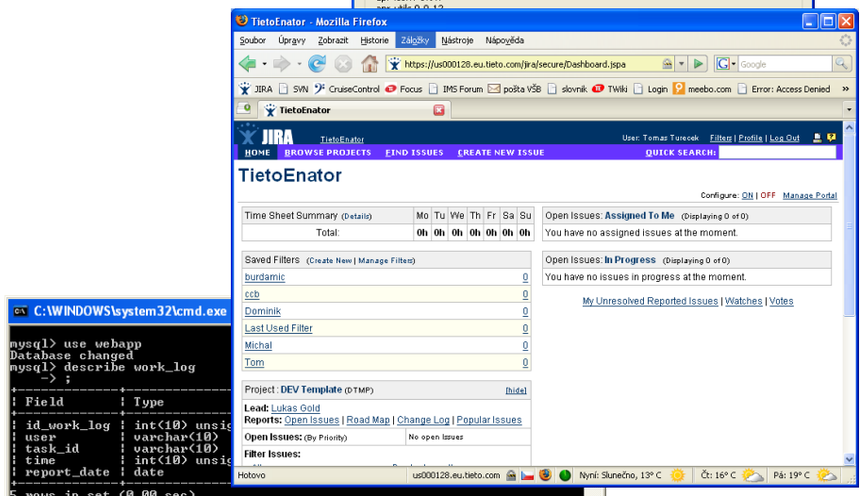

Click to return to the Dashboard.
Click to list projects and their issues.
Click to search for issues and display them in the Issue Navigator.
Click to create a new issue in a project of your choice.
Your name shown here indicates that you are currently logged in to JIRA.
Click to list your saved issue filters.
Click to choose your language or change your password.
Click to print this page.
Click for help about this page.
Type your Quick Search criteria here, then press 'Enter'.
1
2
3
JIRA
User: Sally Jones
Filters | Profile | Log Out
HOME
BROWSE PROJECTS
FIND ISSUES
CREATE NEW ISSUE
QUICK SEARCH:
My Company's JIRA
Configure: ON | OFF Manage Portal
Project : ABC (ABC)
[hide]
Lead: Mary Smith
Reports: Open Issues | Road Map | Change Log | Popular Issues
Open Issues: (By Priority)
Filter Issues:
- All
- Outstanding
- Unscheduled
- Assigned to me
- Reported by me
- Resolved recently
- Added recently
- Updated recently
- Most important
Saved Filters (Create New | Manage Filters)
You have no saved filters at the moment. Create new filters.
Open Issues: Assigned To Me (Displaying 2 of 2)
ABC-3 Change colour of strawberries to red
ABC-2 Change colour of banana to yellow
Open Issues: In Progress (Displaying 0 of 0)
You have no issues in progress at the moment.
My Unresolved Reported Issues | Watches | Votes

Prostředí

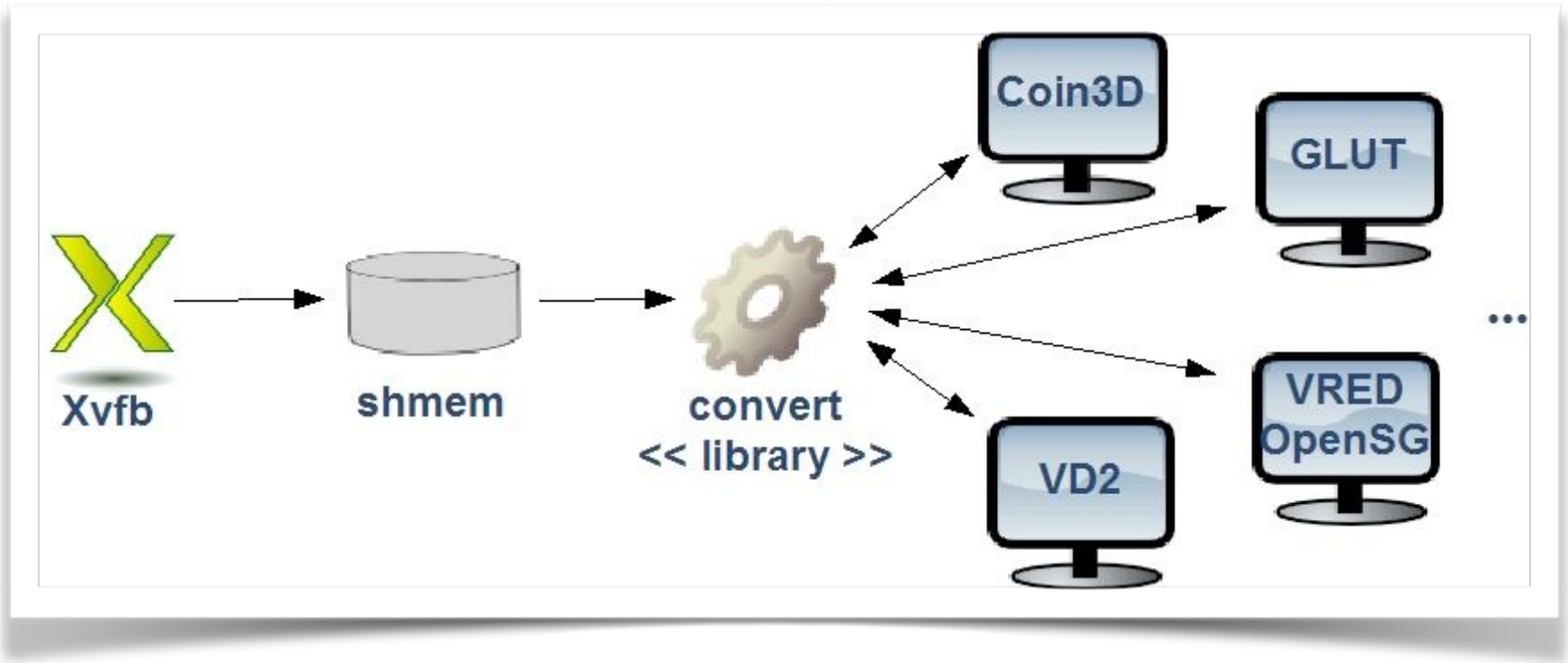
Příklad:

- Java EE
- Eclipse 3.3 + J2EE plugin
- Apache Tomcat 6.0
- MySQL DB
- JDBC driver
- Subversion - Repository
- Jira - Issue tracking tool





Deployment



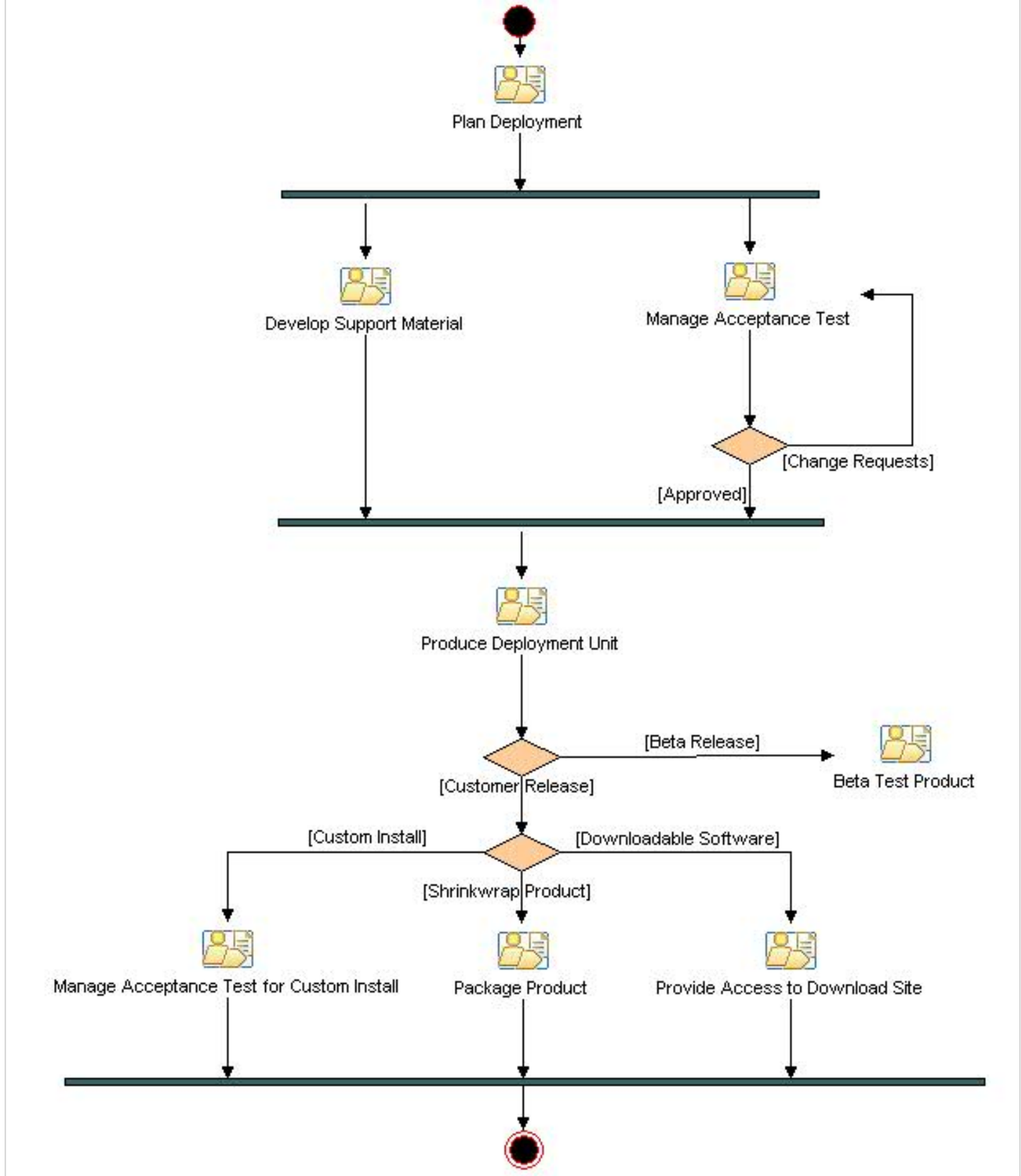
Deployment

Cílem disciplíny je sestavit, testovat a nasadit aplikaci nejen pro interní potřeby týmu (ověření kvality), ale hlavně u zákazníka.

To zahrnuje:

- Plánování nasazení (velmi brzy na začátku životního cyklu)
- Tvorbu podpůrných a systémových a tréninkových materiálů
- „Balení“ a doručení nezbytných artefaktů (servery, release notes, dokumentace)
- Testování ve vývojovém (testovacím) prostředí
- Beta testování, akceptační testování (zákazník)

Disciplína vstupuje do hry **vždy** při každém doručení buildu (nějaké části aplikace) zákazníkovi.


Plan Deployment
Develop Support Material
Manage Acceptance Test
[Change Requests]
[Approved]
Produce Deployment Unit
[Beta Release]
[Customer Release]
Beta Test Product
[Custom Install]
[Downloadable Software]
[Shrinkwrap Product]
Manage Acceptance Test for Custom Install
Package Product
Provide Access to Download Site

Aktivity

- **Plánování nasazení** – řízeno přáním zákazníka, informace k dispozici (musí vědět doručení). Kromě pravidelných demonstrací na konci iteraci a těsné kooperaci s analytiky probíhá beta testování v rámci několika posledních iterací.
- **Tvorba podpůrných materiálů** – veškeré materiály potřebné instalaci produktu, jeho provozu, užívání a údržbě, včetně tréninkových materiálů.
- **Tvorba releasů** – instalační balíček obsahující všechny potřebné artefakty ve správných verzích.

Další aktivity jsou Beta test release, balíčkování produktu, poskytnutí přístupu k webu včetně podpory 24x7.

Úkoly jednotlivých rolí

- **Deployment manager** – plánuje a organizuje nasazení, zajišťuje a kontroluje vhodnou strukturu balíčků.

- **Project manager** – hlavní rozhraní mezi vývojovým týmem a zákazníkem, odpovědný za schvalování nasazení na základě zpětné vazby (neúplné, netestované či zákazníkem neschválené funkčnosti nemohou být instalovány).

- **Tvůrce technické dokumentace** – plánuje a píše uživatelskou dokumentaci a dokumentaci pro podporu koncových uživatelů.

- **Configuration manager** a administrátorské role

- **Vývojář** – vytváří implementační skripty a podpůrné artefakty, které pomáhají uživatelům při instalaci.

- Grafik

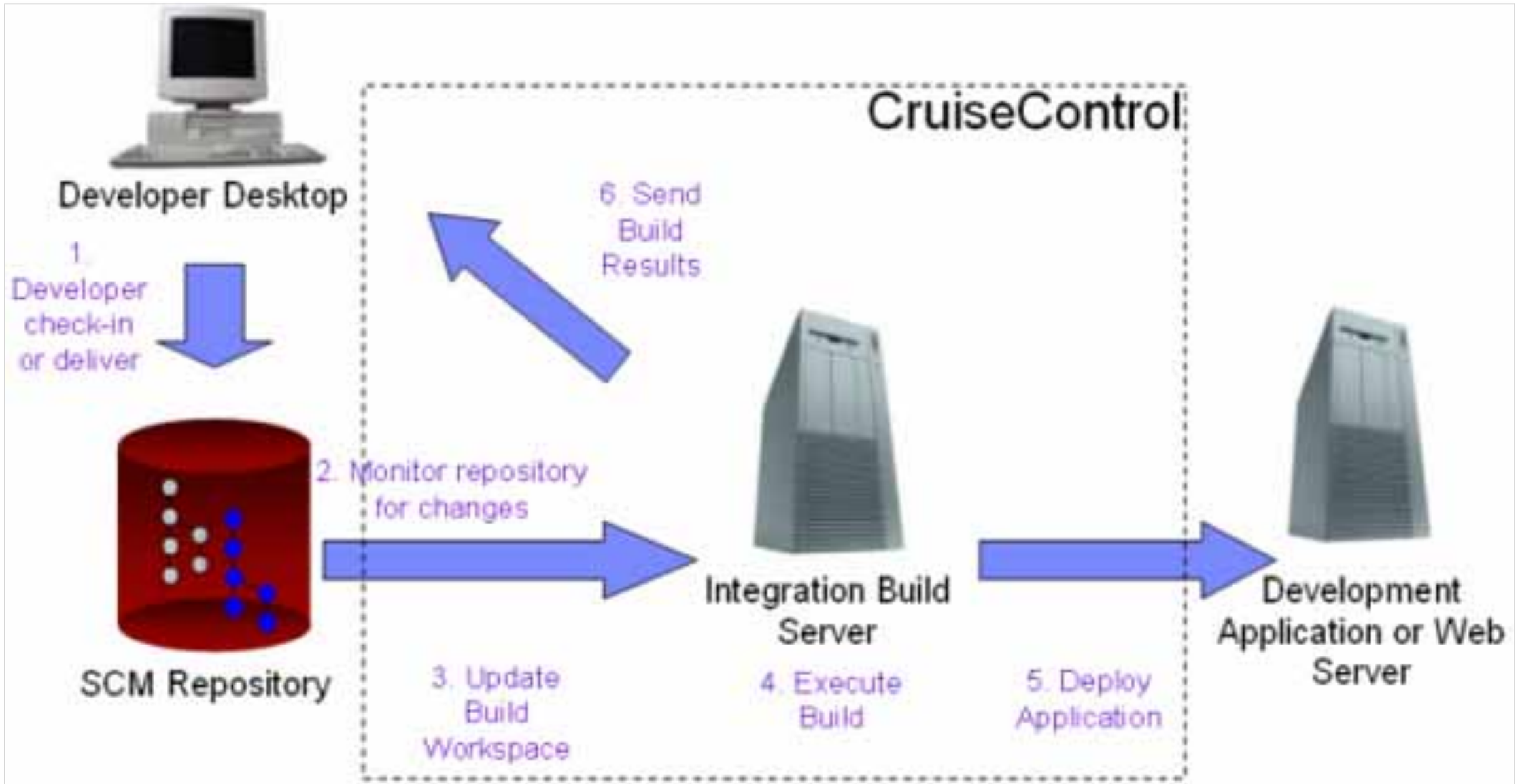
- **Analytik testů**, **tester**

Příklad - technical writer

Artefakty releasu

- **Spustitelný software**,
- **Instalační artefakty** – skripty (např. DB), nástroje, soubory, licenční informace, instalační průvodce,
- **Release notes** popisující release pro koncového zákazníka (+nové funkce, známé chyby, …),
- **Podpůrné materiály** jako provozní či uživatelské příručky a manuály,
- **Tréninkové materiály**.

Continuous integration II.



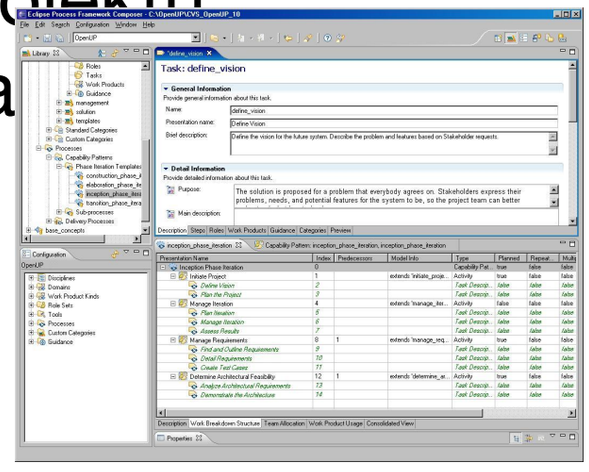
Environment

Environment

- Poskytnutí prostředí vývojové organizace pro podporu vývojového týmu
  - Procesy
  - Nástroje
- Prostředí vytvořeno v prvních iteracích projektu
- V průběhu dalších iterací je prostředí propracováno, upraveno dle potřeb projektu

Hlavní úkoly disciplíny

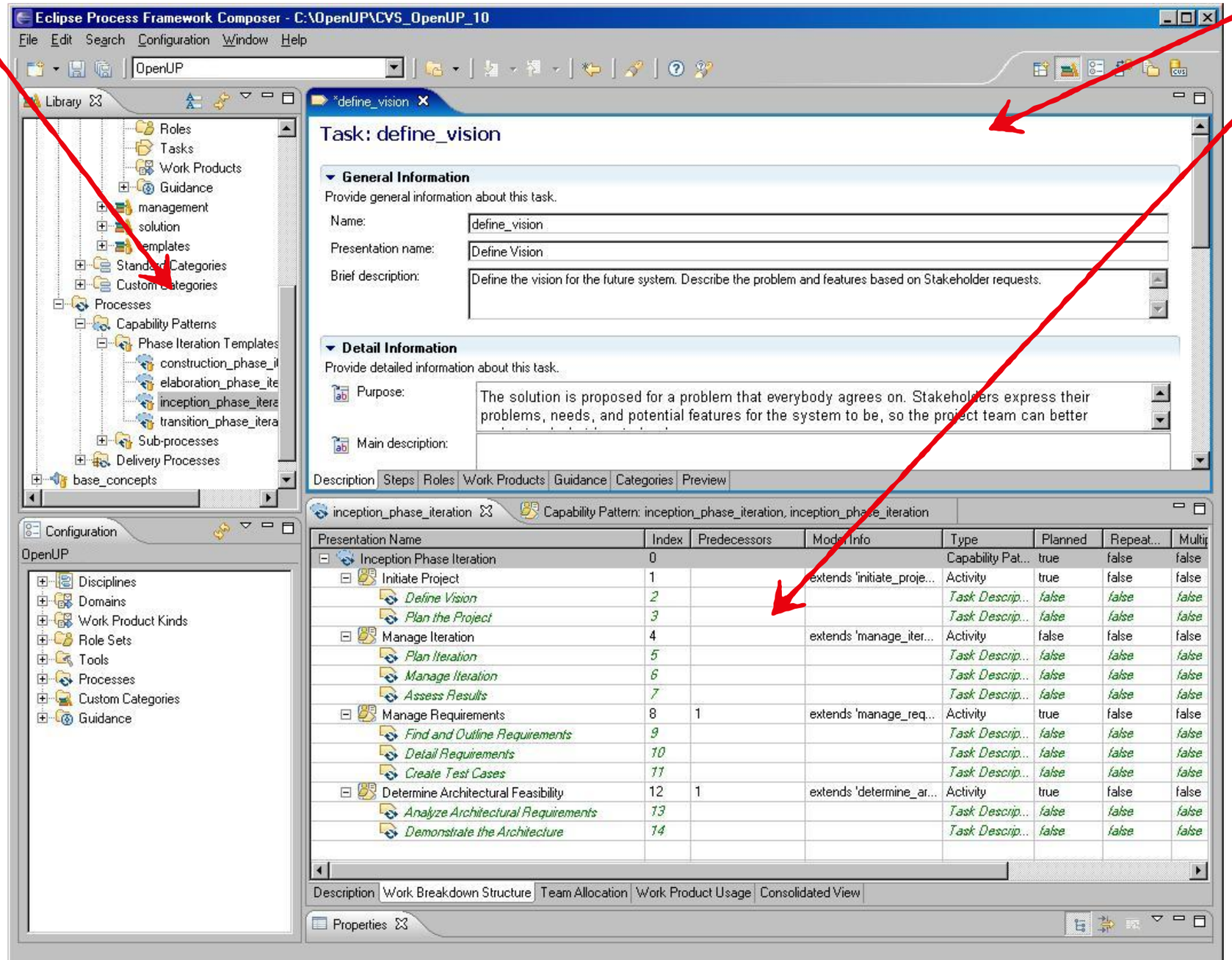
- Definice popisu stylů pro tvorbu příruček, průvodců, šablon.
- Konfigurace procesu – uvedení vývojového procesu do projektu (úprava podle potřeb, publikace procesu, tréninky, podpůrné materiály) a jeho aktualizace, úpravy a zlepšování podle potřeby (retrospektiva).
- Příprava vlastních šablon a průvodců procesu.
- Výběr a zajištění (nákup) nástrojů pro potřeby projektu případně vývoj vlastních nástrojů či jejich integra
- Instalace a nastavení nástrojů.



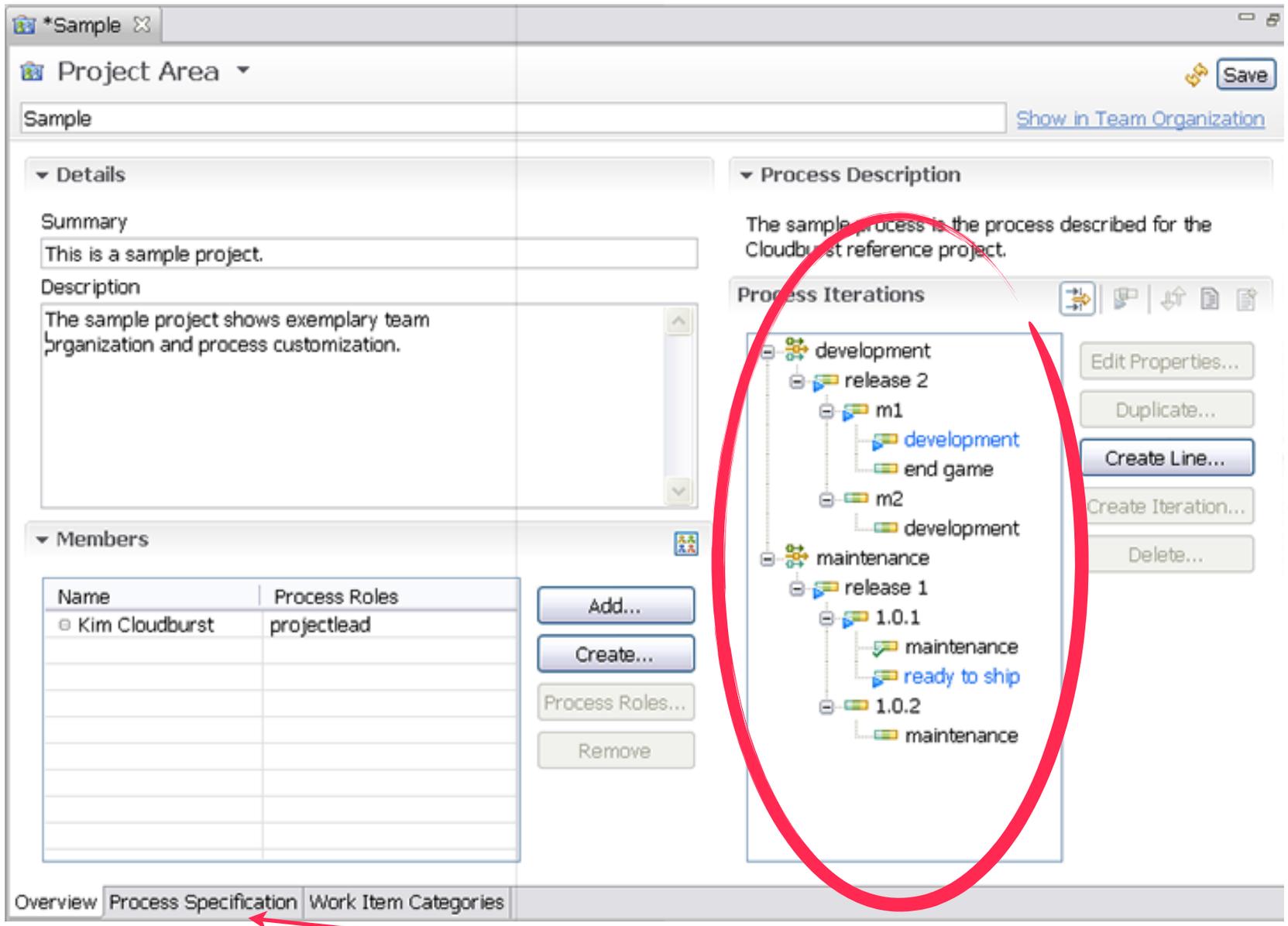
EPF Composer

Artefakty procesu

Detaily artefaktu

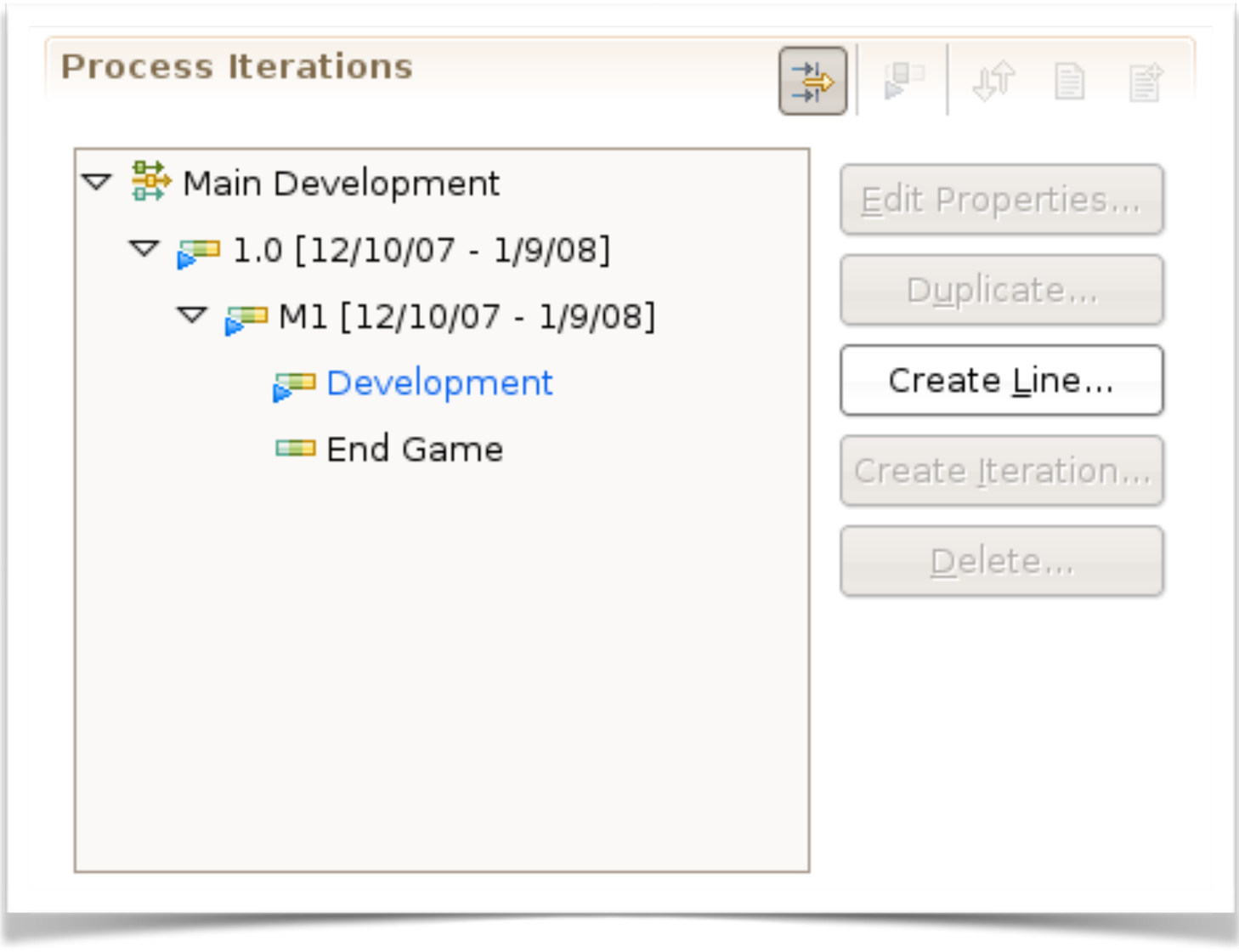


IBM Jazz + RTC



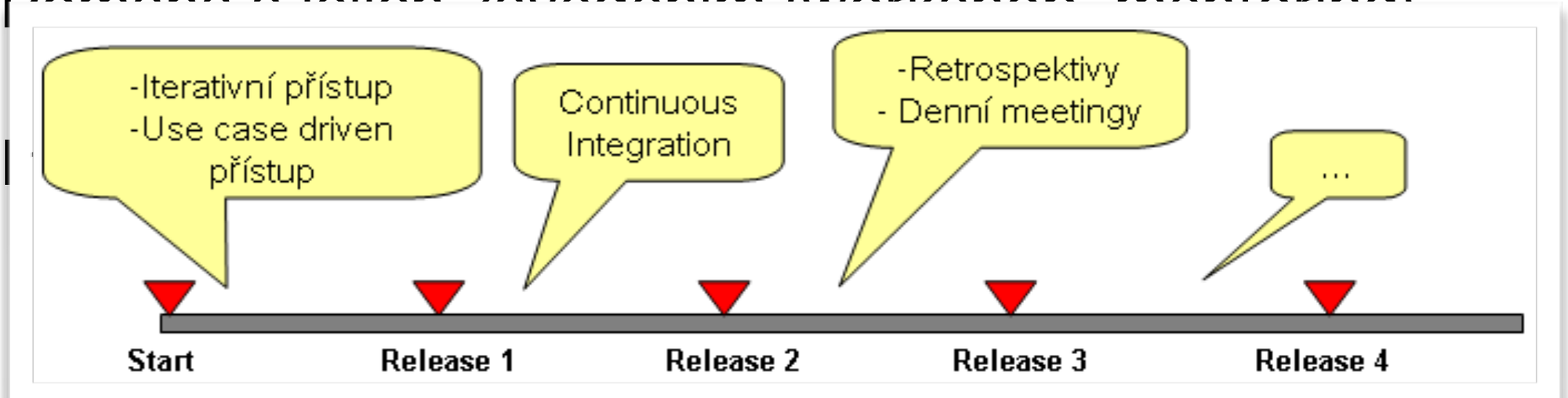
Definice procesu

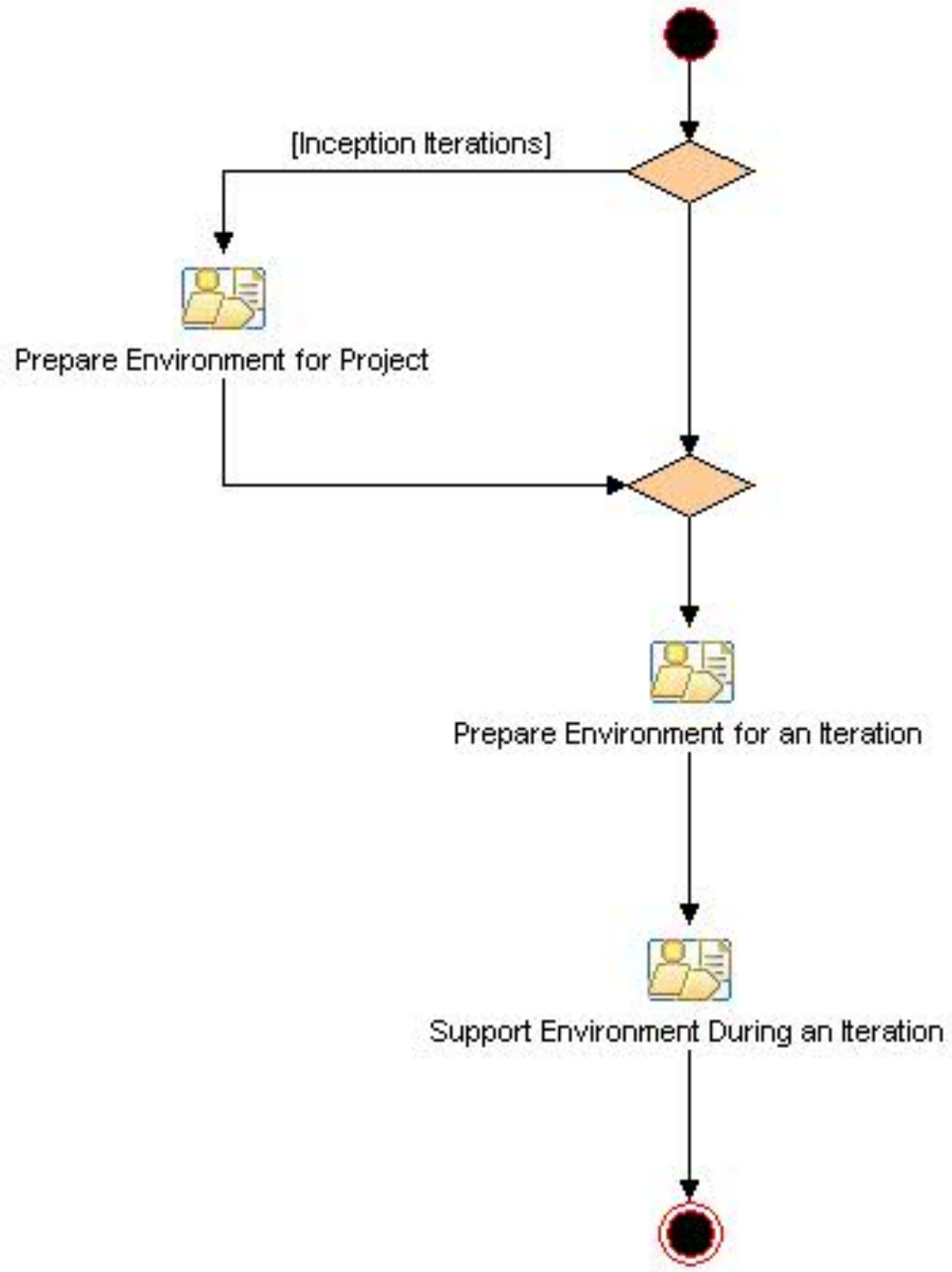
IBM Jazz + RTC


Process Iterations
Main Development
1.0 [12/10/07 - 1/9/08]
M1 [12/10/07 - 1/9/08]
Development
End Game
Edit Properties...
Duplicate...
Create Line...
Create Iteration...
Delete...

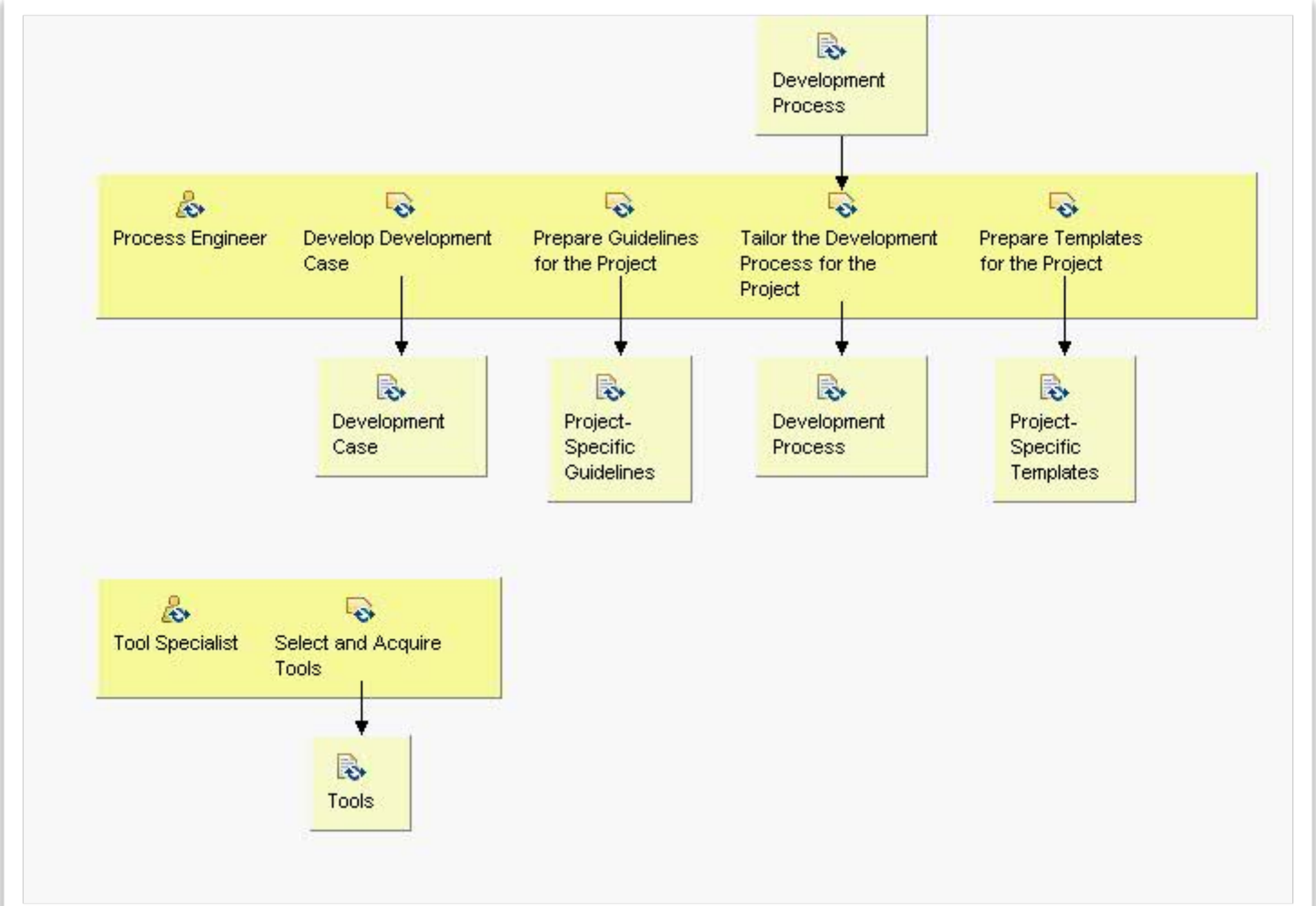
Mentor

- Zkušená osoba
- Upravuje proces
- Definuje vhodné techniky
- Pomáhá s jejich zavedením (workshop, mentoring)
- Kontroluje




[Inception Iterations]
Prepare Environment for Project
Prepare Environment for an Iteration
Support Environment During an Iteration

Role


Development Process
Process Engineer
Develop Development Case
Prepare Guidelines for the Project
Tailor the Development Process for the Project
Prepare Templates for the Project
Development Case
Project-Specific Guidelines
Development Process
Project-Specific Templates
Tool Specialist
Select and Acquire Tools
Tools

Nástroje

IDE – integrované vývojové prostředí

- Eclipse, NetBeans, …
- Visual Studio

Nástroj pro správu požadavků, chyb

- Rational Requisite Pro, Jira, …
- Rational Team Concert

Nástroje pro správu konfigurací a změn – Configuration and Change Management Tools

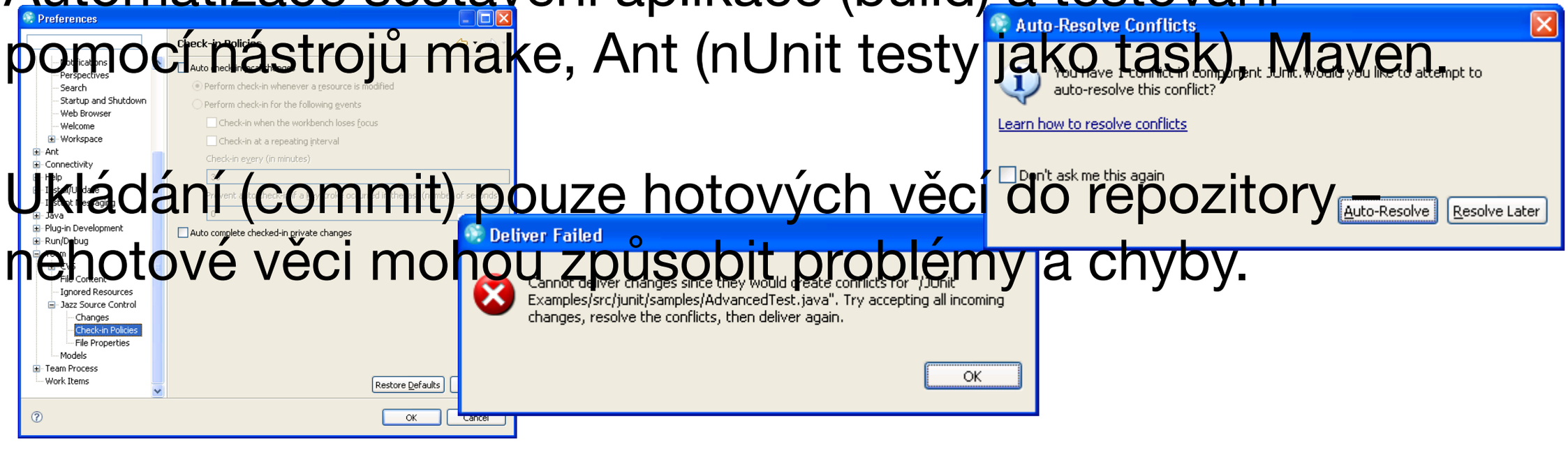
- **CVS,**
- **SVN,**
- **Jira**

Vizuální modelovací nástroje

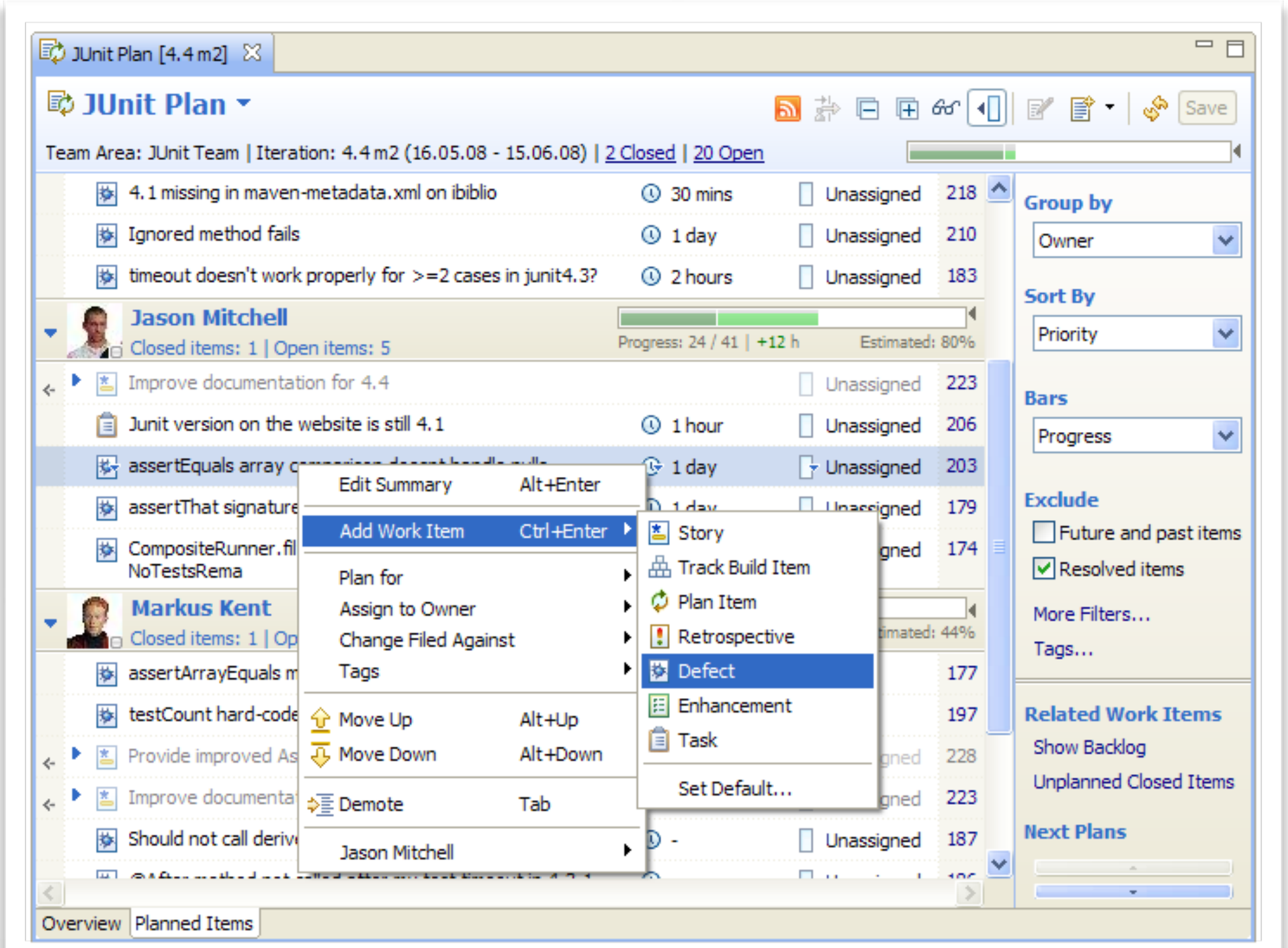
- **Magic Draw,**
- **Borland Together,**
- **Eclipse pluginy**

Nástroje a tým

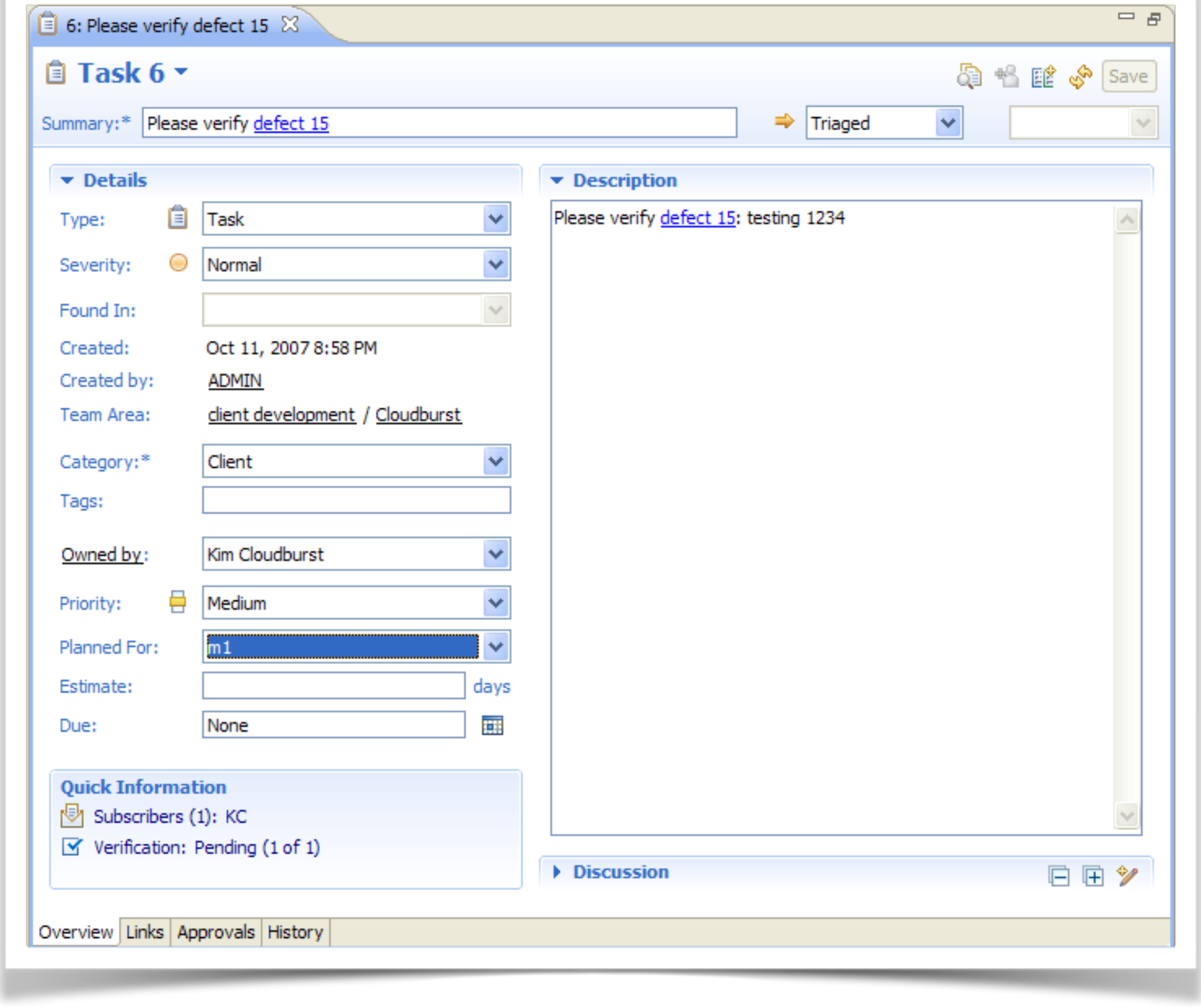
- Týmová práce, neustálá spolupráce různých rolí
- Úložiště zdrojových kódů a dokumentace (repozitory) – cílem by mělo být dosáhnout možnosti sestavení kompletní aplikace z repozitory.
- Automatizace sestavení aplikace (build) a testování – pomocí nástrojů make, Ant (nUnit testy jako task), Maven.
- Ukládání (commit) pouze hotových věcí do repozitory – nehotové věci mohou způsobit problémy a chyby.



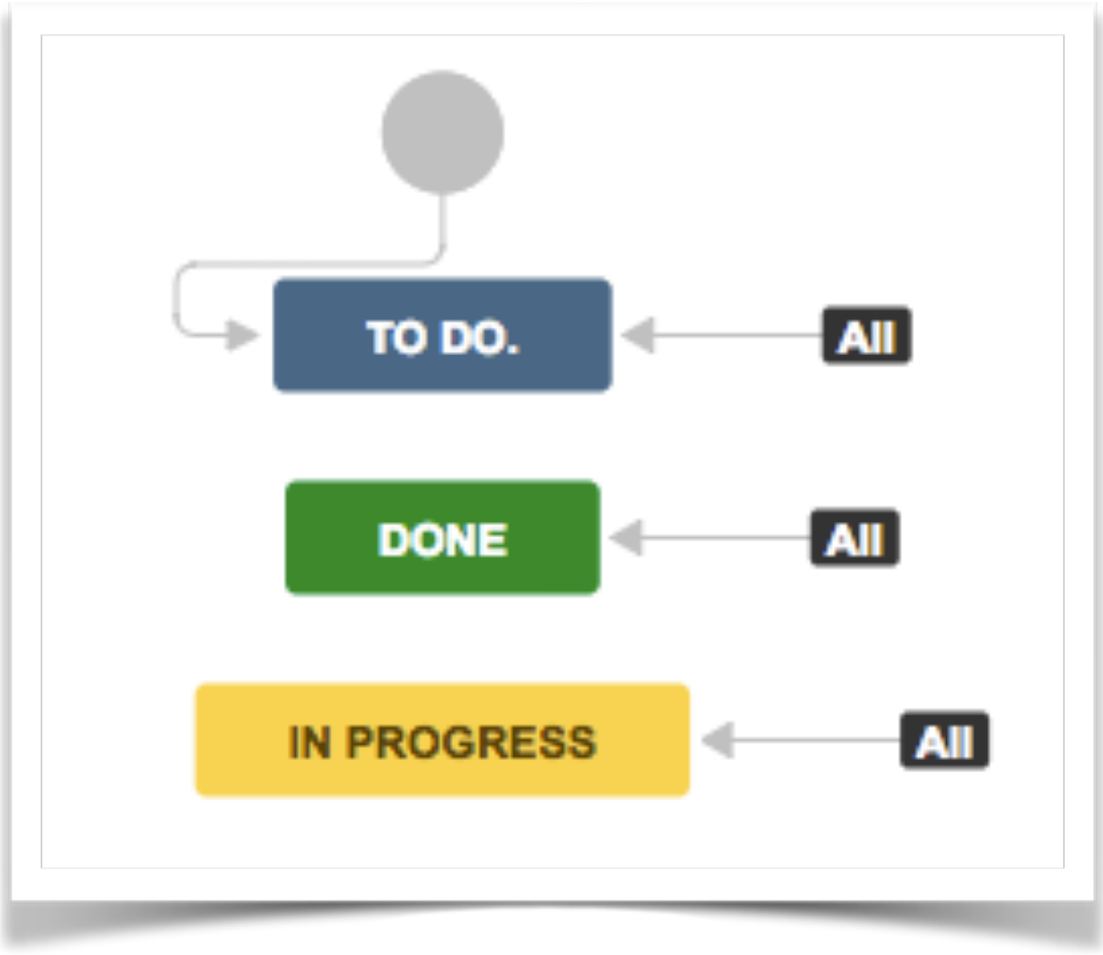
IBM RTC - týmový pohled



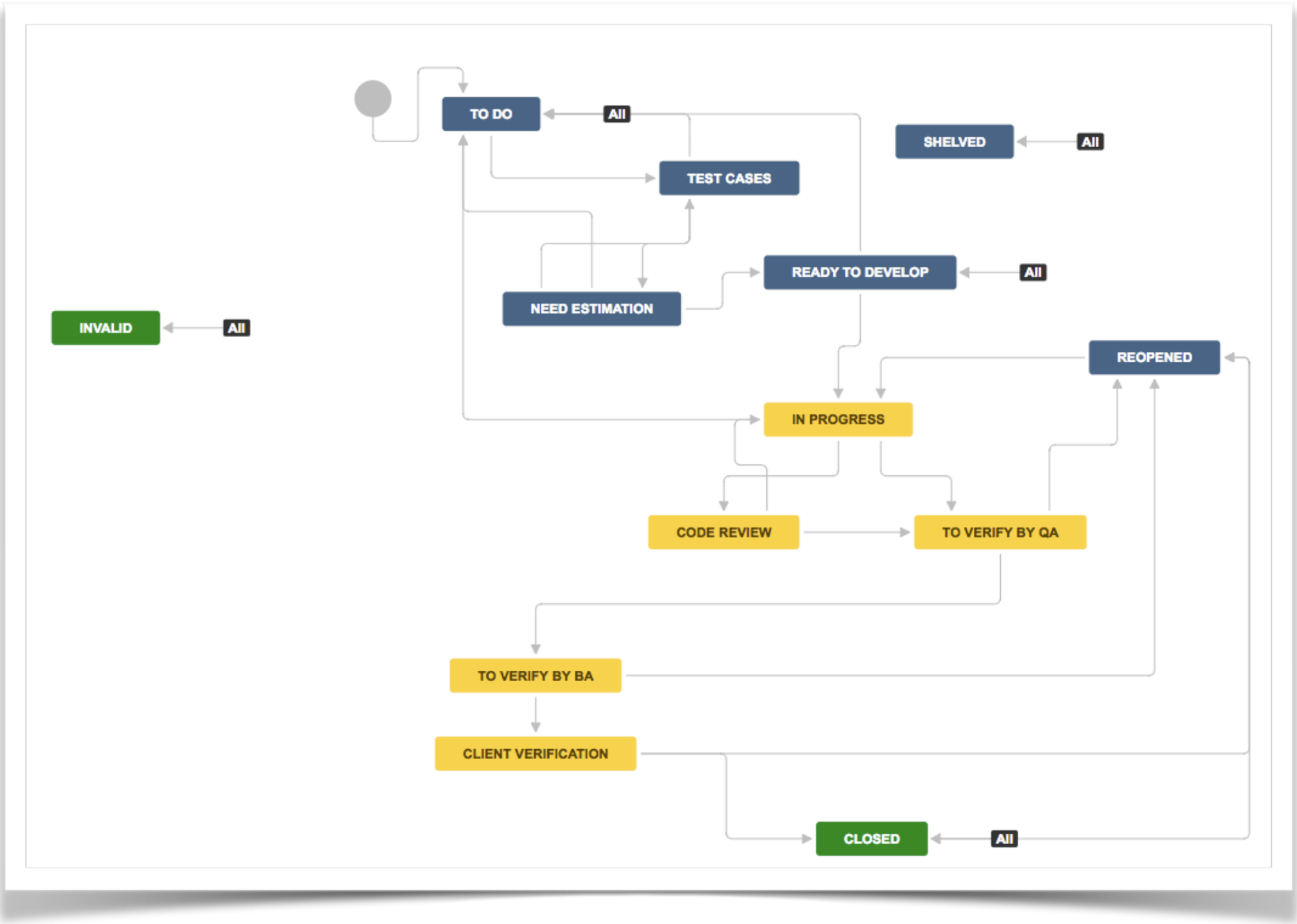
RTC - detail úkolu



Jira proces



Jira proces

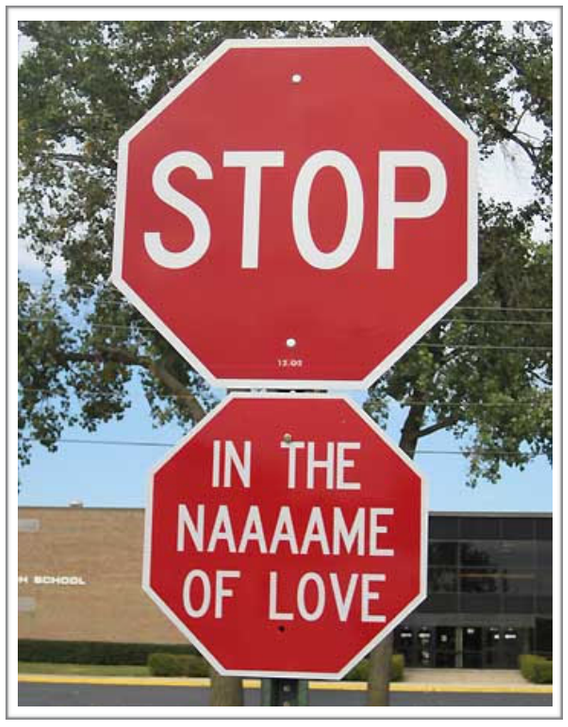

TO DO
All
SHELVED
All
TEST CASES
READY TO DEVELOP
All
NEED ESTIMATION
INVALID
All
REOPENED
IN PROGRESS
CODE REVIEW
TO VERIFY BY QA
TO VERIFY BY BA
CLIENT VERIFICATION
CLOSED
All

Kombinace rolí

Jak kombinovat role

NE: Technické + Netechnické

- analytik + vývojář
- architekt + tester
- vývojář + tester



ANO analytik + tester (doporučeno)

ANO manažerské role (PM, CM, …)

ANO architekt + vývojář

ANO vývojář + CM

ANO architekt + procesní mentor