Vývoj Aplikácií s Viacvrstvovou Architektúrou

03. Vývojové Technológie a Biznis Vrstva

IT ACADEMY

STU FIIT

Lektor: PhDr. Ing. Mgr. Miroslav Reiter, DiS., MBA, MPA
Kontakt: riaditel@it-academy.sk

Čo nás čaká a neminie...

1. časť

- Úvod do Javy
- Štruktúra Platformy a EA
- Vývojové Technológie (BIZ)
- Kolekcie (APP)
- Logovanie
- Lokalizácia

2. časť

- JDBC a DBMS (TECH)
- XML, NIO2
- Regulárne Výrazy
- Prehľad EA
- Támec TOGAF a ADM
- Prehľad JEE a .NET

Best practices | Faily

Čo nás čaká a neminie...

1. časť

Architektúra

2. časť

Java

Best practices | Faily

Vývoj Aplikácií s Viacvrstvovou Architektúrou

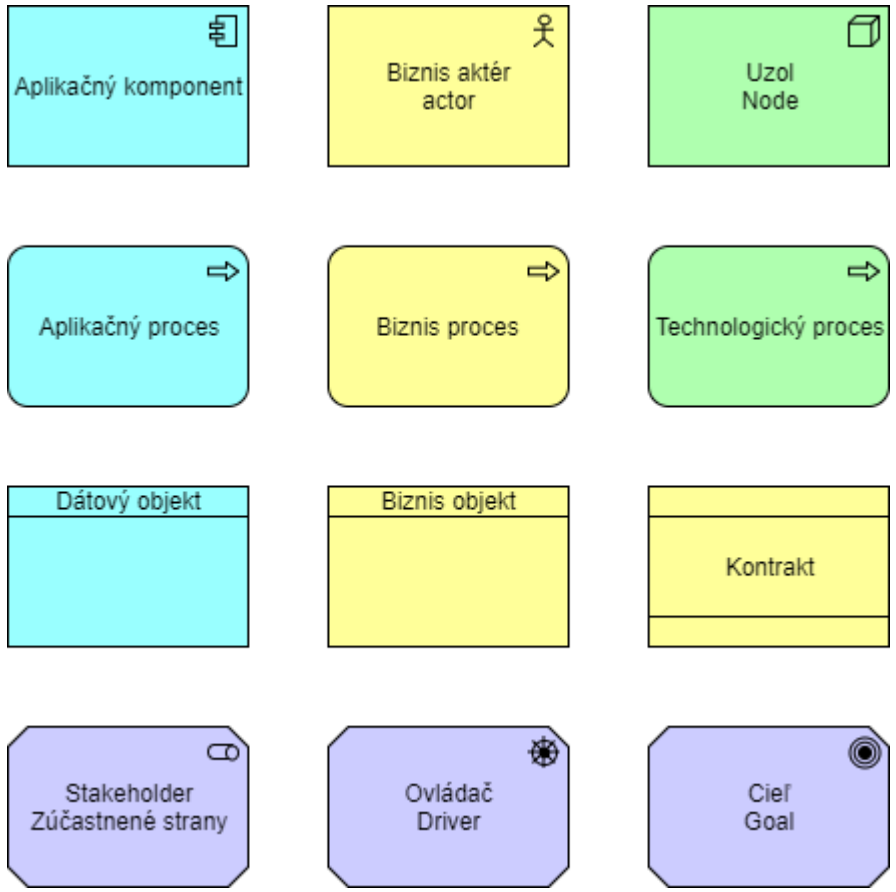
03. Vývojové Technológie a Biznis

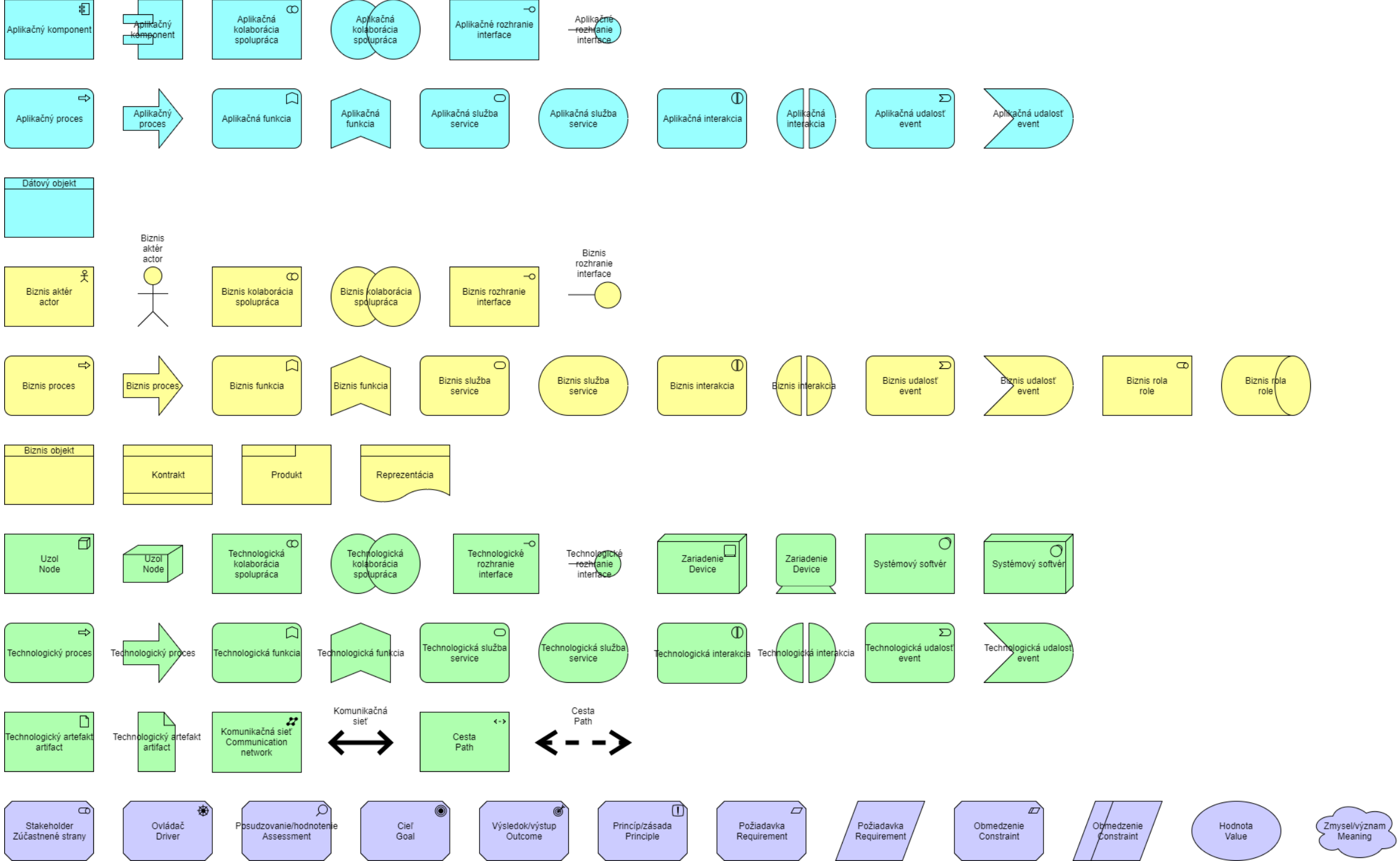
STU FIIT

Lektor: **PhDr. Ing. Mgr. Miroslav Reiter, DiS., MBA, MPA**
Kontakt: riaditel@it-academy.sk

Rozlíšenie elementov podľa rohov

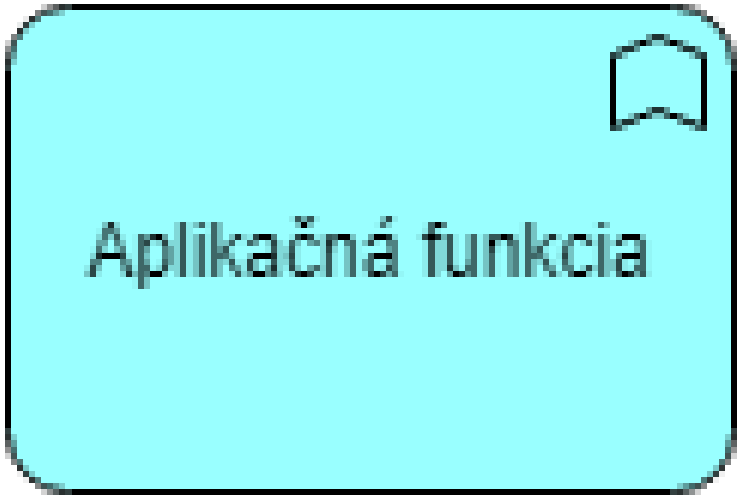
1. Ostré rohy – aktívne elementy
2. Zaoblené rohy – element chovania (behavior)
3. Ostrý roh s preškrtnutím – pasívne elementy
4. Skosené rohy – motivačný element




Aplikačný komponent
Aplikačný komponent
Aplikačná kolaborácia spolupráca
Aplikačná kolaborácia spolupráca
Aplikačné rozhranie interface
Aplikačné rozhranie interface
Aplikačný proces
Aplikačný proces
Aplikačná funkcia
Aplikačná funkcia
Aplikačná služba service
Aplikačná služba service
Aplikačná interakcia
Aplikačná interakcia
Aplikačná udalosť event
Aplikačná udalosť event
Dátový objekt
Biznis aktér actor
Biznis aktér actor
Biznis kolaborácia spolupráca
Biznis kolaborácia spolupráca
Biznis rozhranie interface
Biznis rozhranie interface
Biznis proces
Biznis proces
Biznis funkcia
Biznis funkcia
Biznis služba service
Biznis služba service
Biznis interakcia
Biznis interakcia
Biznis udalosť event
Biznis udalosť event
Biznis rola role
Biznis rola role
Biznis objekt
Kontrakt
Produkt
Reprezentácia
Uzol Node
Uzol Node
Technologická kolaborácia spolupráca
Technologická kolaborácia spolupráca
Technologické rozhranie interface
Technologické rozhranie interface
Zariadenie Device
Zariadenie Device
Systémový softvér
Systémový softvér
Technologický proces
Technologický proces
Technologická funkcia
Technologická funkcia
Technologická služba service
Technologická služba service
Technologická interakcia
Technologická interakcia
Technologická udalosť event
Technologická udalosť event
Technologický artefakt artifact
Technologický artefakt artifact
Komunikačná sieť Communication network
Komunikačná sieť
Cesta Path
Cesta Path
Stakeholder Zúčastnené strany
Ovládač Driver
Posudzovanie/hodnotenie Assessment
Cieľ Goal
Výsledok/výstup Outcome
Princíp/zásada Principle
Požiadavka Requirement
Požiadavka Requirement
Obmedzenie Constraint
Obmedzenie Constraint
Hodnota Value
Zmysel/význam Meaning

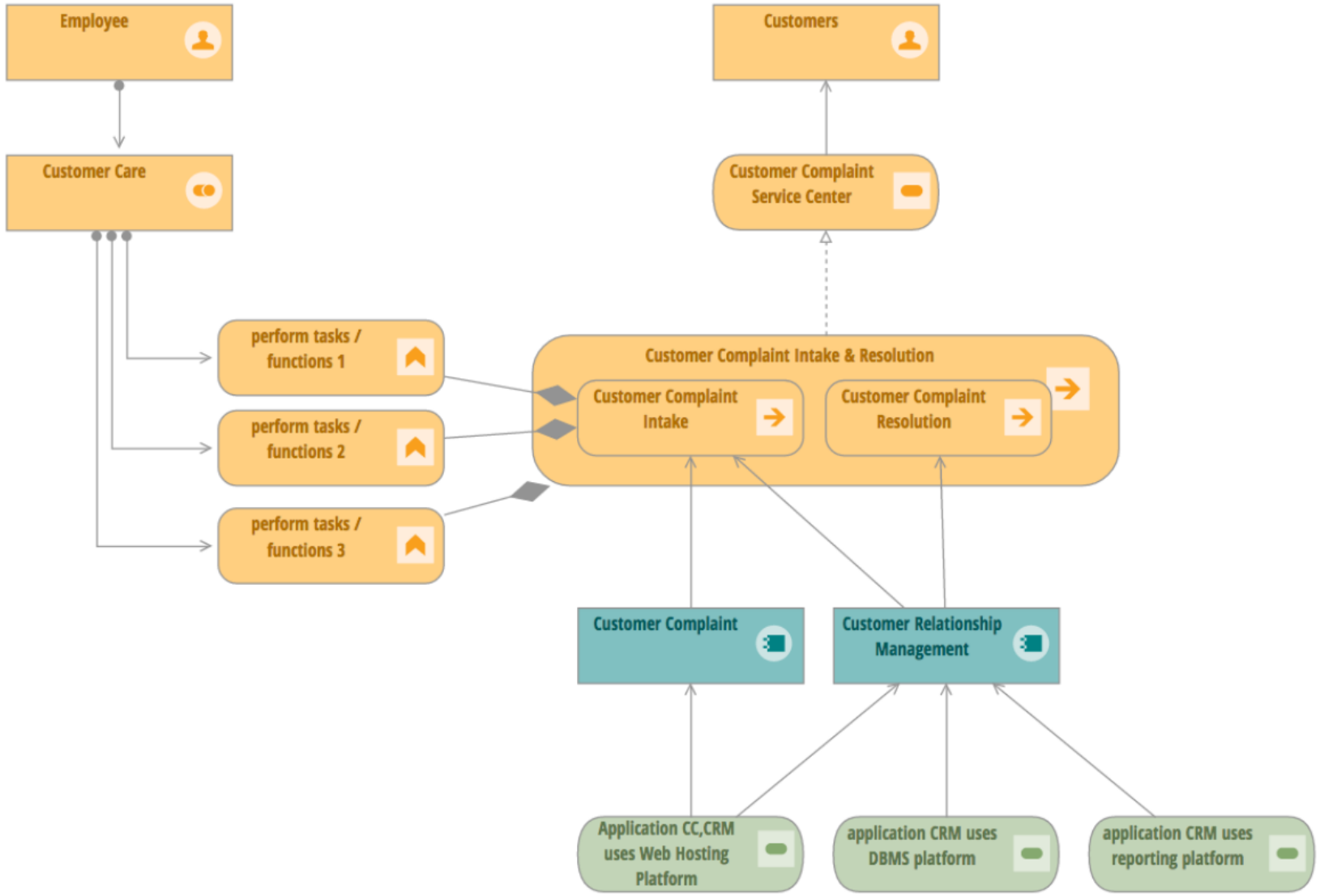
Spôsob zápisu elementov

Obdĺžnikový zápis

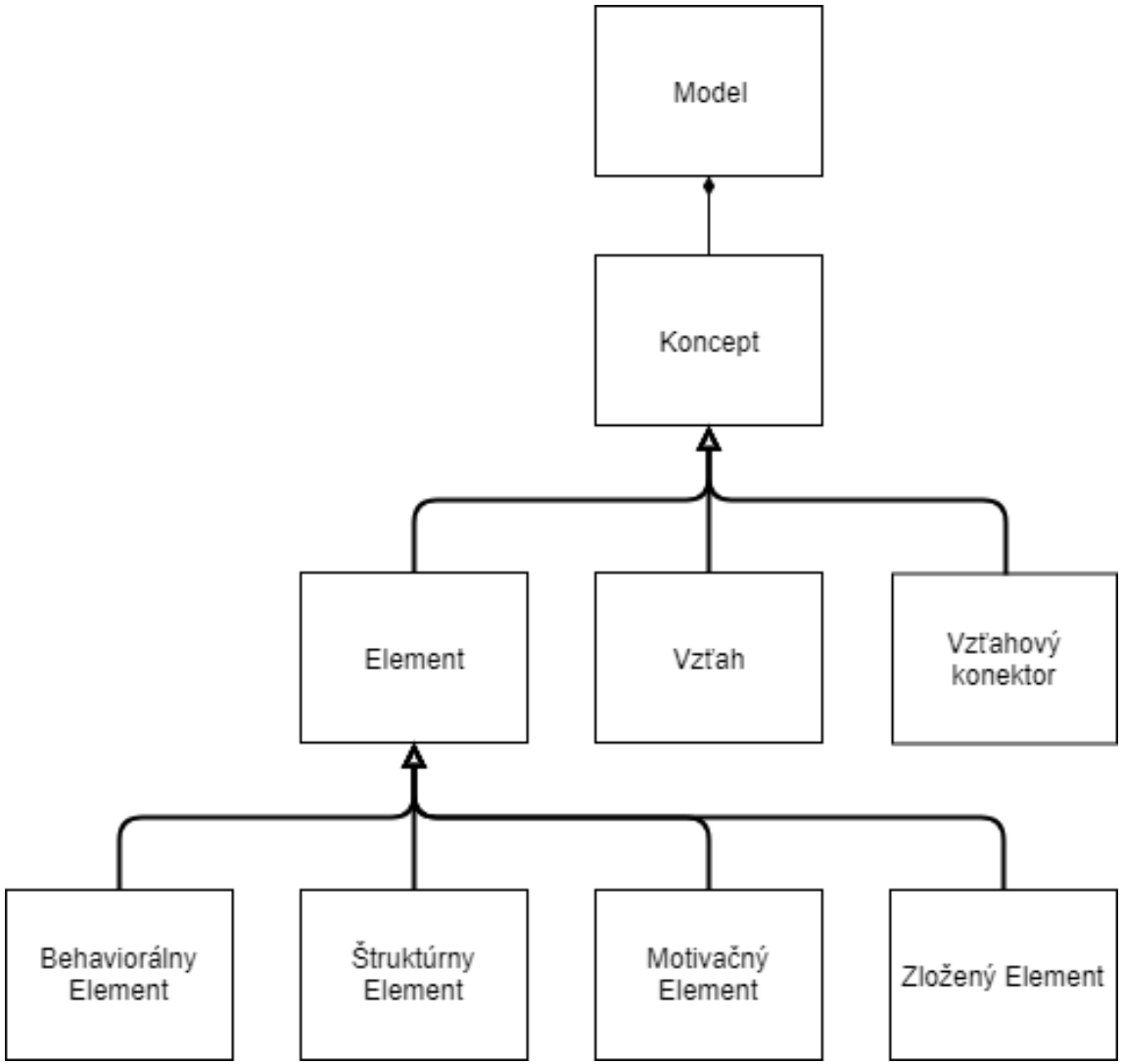


Zápis ikony

Príklad ArchiMate modelu - Core Layers

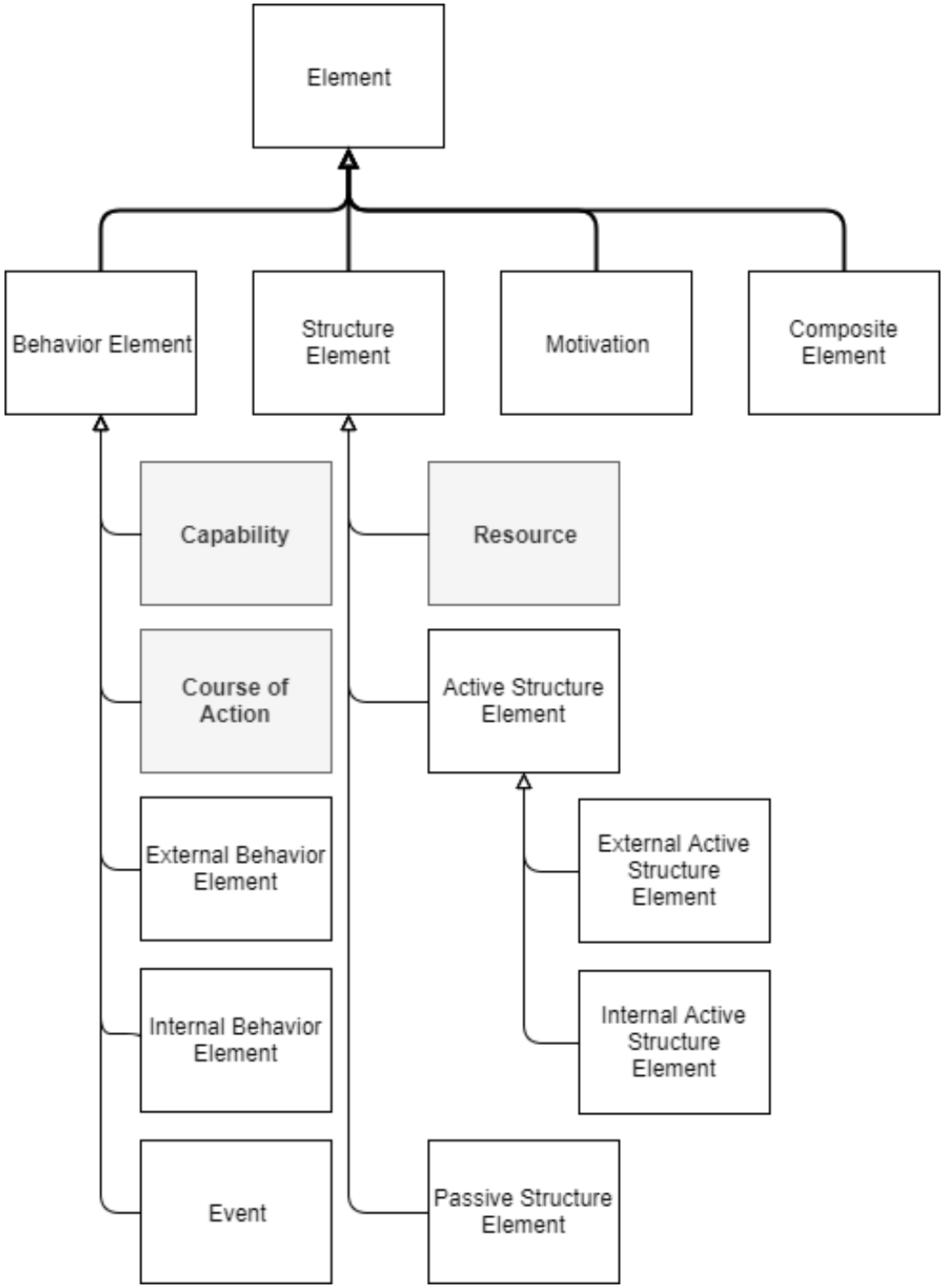


Top Level Jazyková Štruktúra

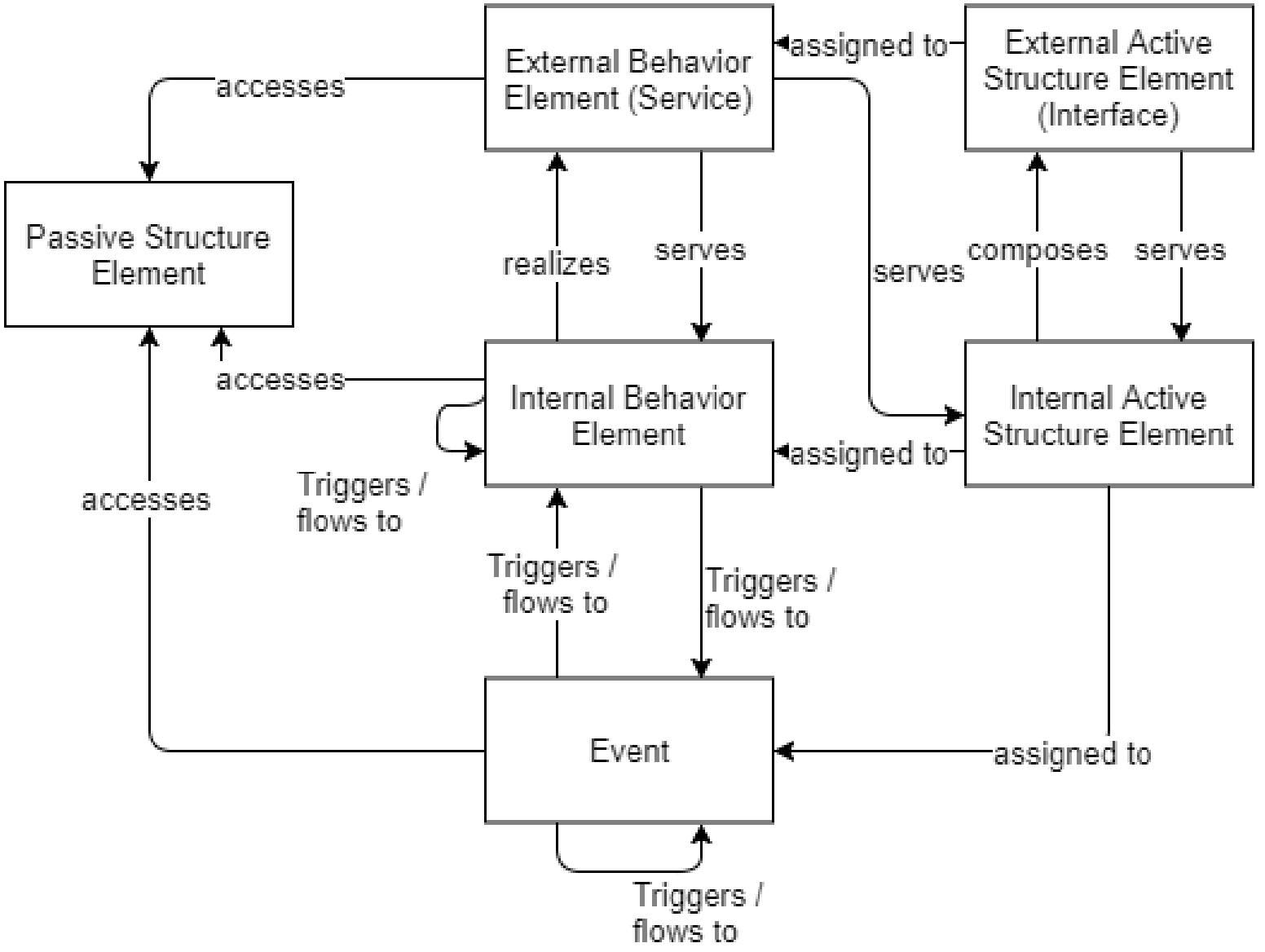


- **Model** je **súborom konceptov**
- **Koncepcia** je buď **element** alebo a **vzťah**
- **Elementom** je buď **prvok správania**, **štruktúrny**, **motivačný** alebo **kompozitný (zložený) element**

ArchiMate Metamodel 1

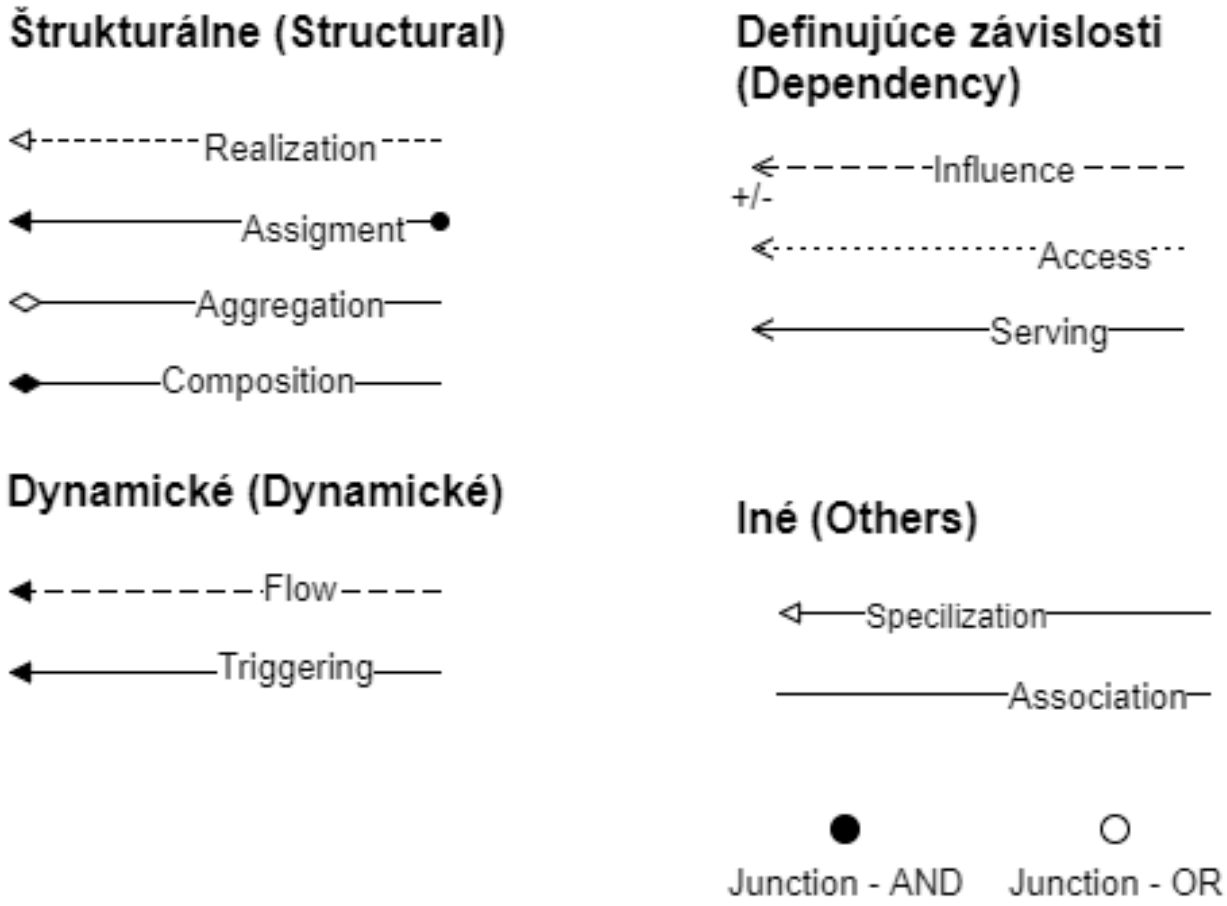


ArchiMate Metamodel 2



Prehľad väzieb (Relationship)

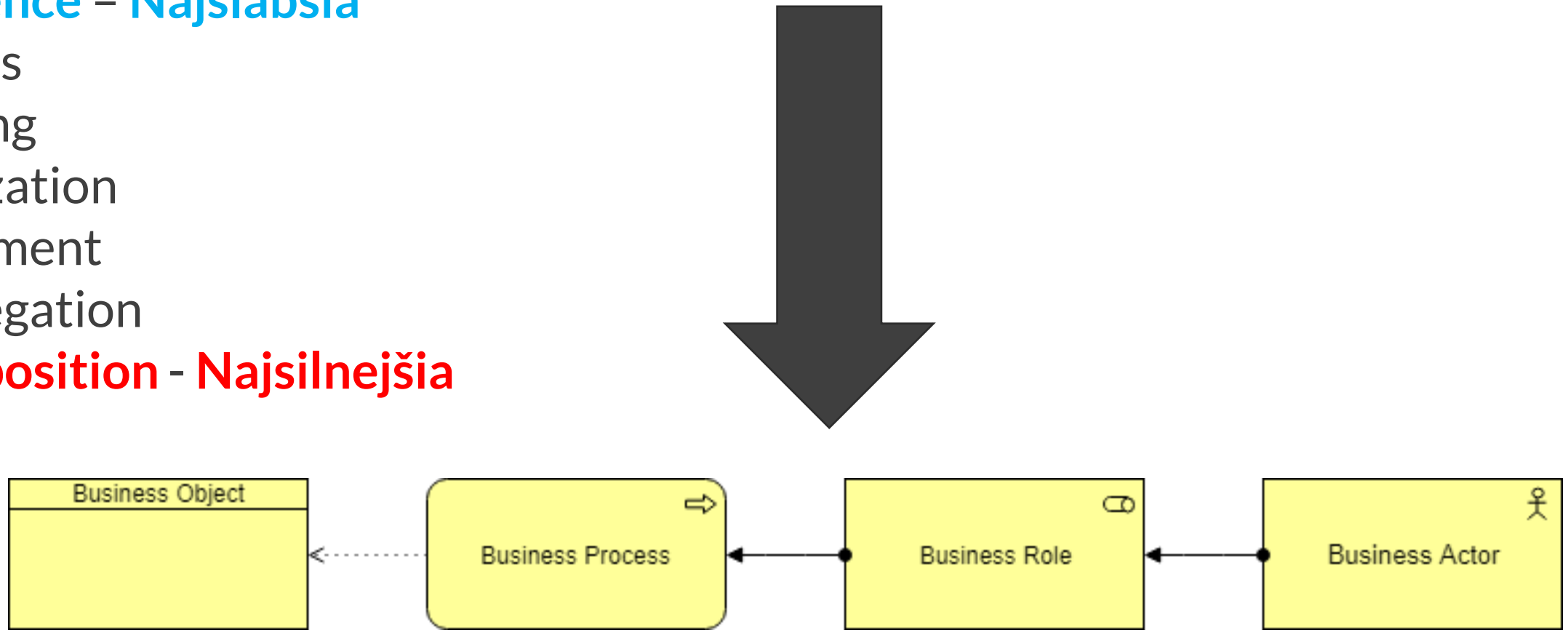
- **Väzby** sa delia **4 skupiny**
- Väzby v jednotlivých kategóriách sú **zoradené** od **najslabšej** po **najsilnejšiu**



Prehľad Väzieb (Relationship)

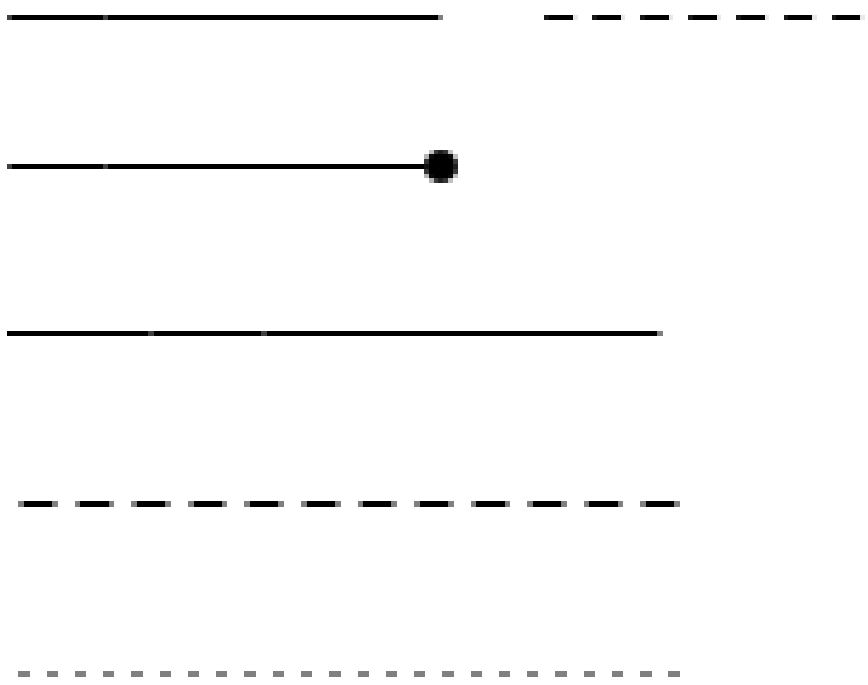
Väzby môžu byť zoradené podľa svojej sily (od najslabšej po najsilnejšiu)

1. **Influence** – **Najslabšia**
2. Access
3. Serving
4. Realization
5. Assigment
6. Aggregation
7. **Composition** - **Najsilnejšia**



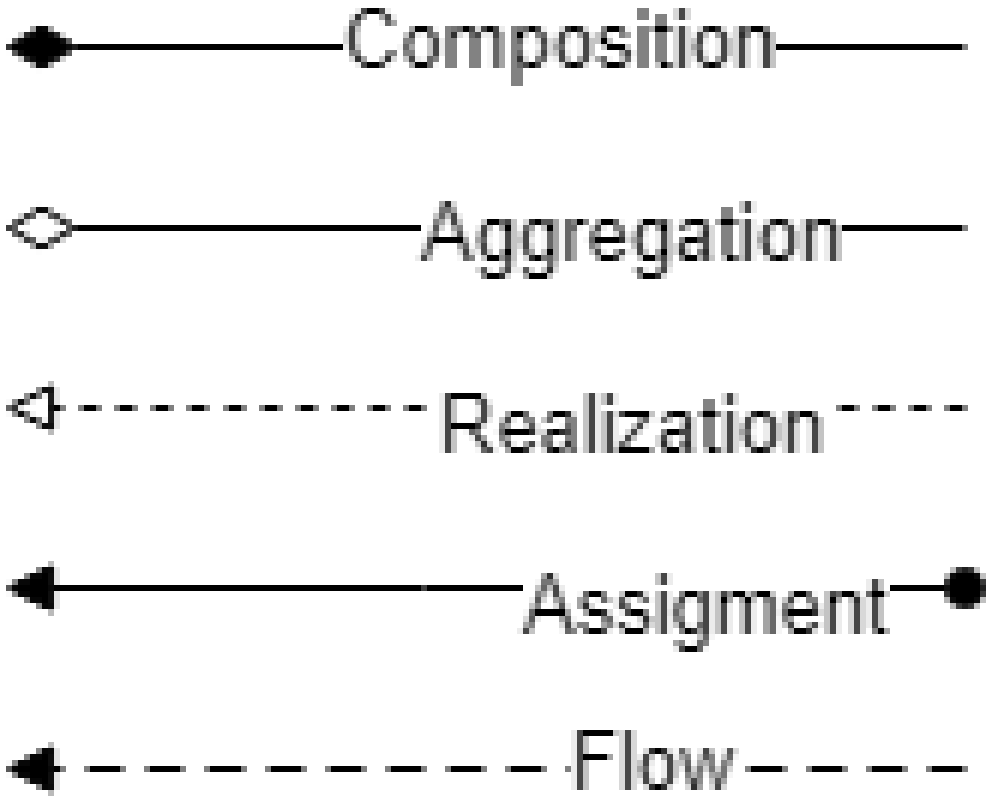
Rozpoznávacie Znaky Väzieb

- **Počiatok väzby**
  - Jednoduchá čiara podľa typu väzby
  - Vyplnený kruh resp. guličkа
- **Telo väzby**
  - Plná čiara
  - Prerušovaná
  - Drobno Prerušovaná

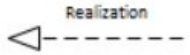
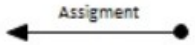
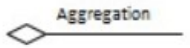
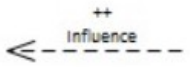


Rozpoznávacie Znaky Väzieb

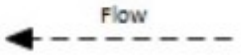
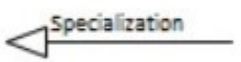
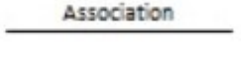
- **Šípka**
  1. Diamant vyplnený
  2. Diamant prázdny
  3. Šípka uzavretá, plná čiara šípky, nevyplnené vnútro
  4. Šípka uzavretá, plná čiara šípky, vyplnené vnútro
  5. Šípka otvorená, plná čiara šípky
  6. Šípka otvorená, prerušovaná čiara šípky



Definície Väzieb 1

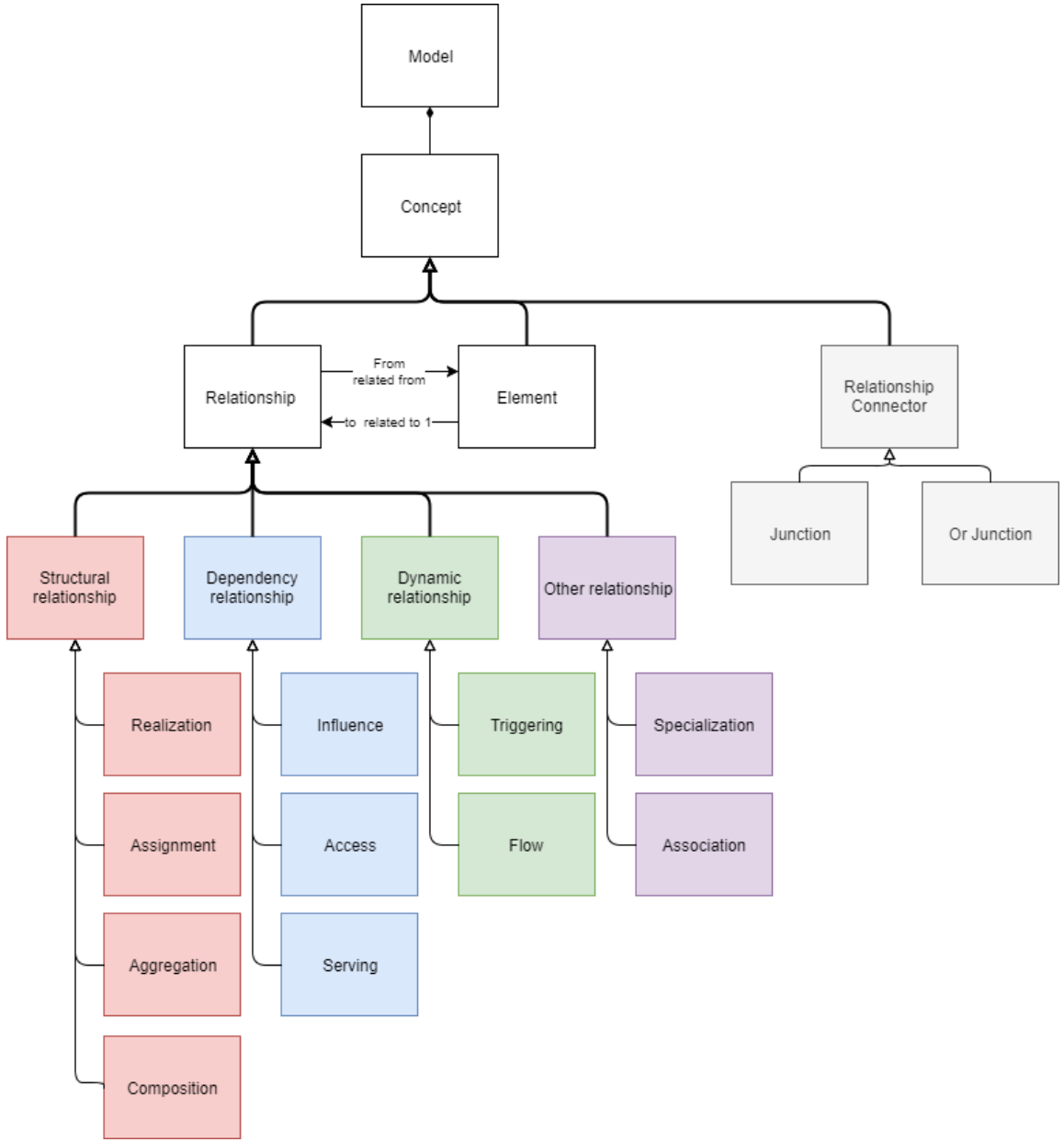
Vazba	Definice	Typ
	The realization relationship indicates that an entity plays a critical role in the creation, achievement, sustenance, or operation of a more abstract entity.	Structural
	The assignment relationship expresses the allocation of responsibility, performance of behavior, or execution.	Structural
	The aggregation relationship indicates that an element groups a number of other elements.	Structural
	The composition relationship indicates that an element consists of one or more other elements.	Structural
	The influence relationship models that an element affects the implementation or achievement of some motivation element.	Dependency
	The access relationship models the ability of behavior and active structure elements to observe or act upon passive structure elements.	Dependency
	The serving relationship models that an element provides its functionality to another element.	Dependency

Definície Väzieb 1

Vazba	Definice	Typ
	The flow relationship represents transfer from one element to another.	Dynamic
	The triggering relationship describes a temporal or causal relationship between elements.	Dynamic
	The specialization relationship indicates that an element is a particular kind of another element.	Other
	An association models an unspecified relationship, or one that is not represented by another ArchiMate relationship.	Other
	A junction is used to connect relationships of the same type.	Other

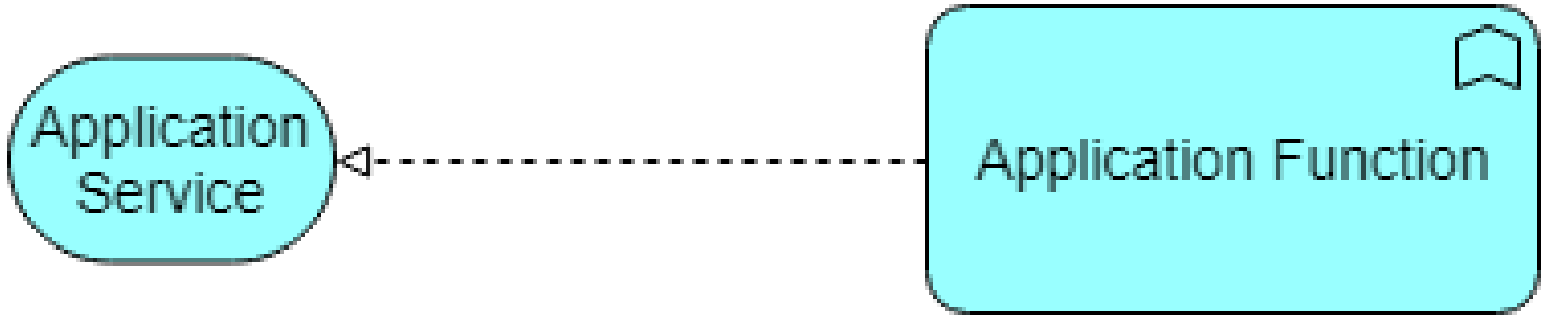
↓ From / → To	Assessment	Constraint	Driver	Goal	Meaning	Outcome	Principle	Requirement	Stakeholder	Value	Capability	Course of Action	Resource	Business Actor	Business Collaboration	Contract	Business Event	Business Function	Business Interaction	Business Interface	Business Object	Business Process	Product	Representation	Business Role	Business Service
Assessment	cgnos	no	no	no	no	no	no	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Constraint	no	cgnos	no	nor	no	nor	nor	cgnos	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Driver	no	no	cgnos	no	no	no	no	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Goal	no	no	no	cgnos	no	no	no	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Meaning	no	no	no	no	cgnos	no	no	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Outcome	no	no	no	nor	no	cgnos	no	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Principle	no	no	no	nor	no	nor	cgnos	no	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Requirement	no	cgnos	no	nor	no	nor	nor	cgnos	o	no	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Stakeholder	o	o	o	o	o	o	o	o	cgnos	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Value	no	no	no	no	no	no	no	no	o	cgnos	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
Capability	no	nor	no	nor	no	nor	nor	nor	o	no	cfgostv	fortv	fot	o	o	o	o	o	o	o	o	o	o	o	o	o
Course of Action	no	nor	no	nor	no	nor	nor	nor	o	no	fot	cfgostv	fot	o	o	o	o	o	o	o	o	o	o	o	o	o
Resource	no	nor	no	nor	no	nor	nor	nor	o	no	fiotv	fortv	cfgost	o	o	o	o	o	o	o	o	o	o	o	o	o
Business Actor	no	nor	no	nor	no	nor	nor	nor	ino	no	or	or	or	cfgostv	fotv	ao	fiotv	fiortv	fiortv	cfgiotv	ao	fiortv	fot	ao	fiotv	fiortv
Business Collaboration	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	or	fgotv	cfgostv	ao	fiotv	fiortv	fiortv	cfgiotv	ao	fiortv	fot	ao	fgiotv	fiortv
Contract	no	nor	no	nor	no	nor	nor	nor	o	no	or	or	or	o	o	cgos	o	o	o	o	cgos	o	o	o	o	o
Business Event	no	nor	no	nor	no	nor	nor	nor	o	no	o	o	o	fot	fot	ao	cfgost	fot	fot	fot	ao	fot	fot	ao	fot	fot
Business Function	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	o	fotv	fotv	ao	fotv	cfgorstv	cfgortv	fotv	ao	cfgortv	fot	ao	fotv	fortv
Business Interaction	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	o	fotv	fotv	ao	fotv	cfgortv	cfgorstv	fotv	ao	cfgortv	fot	ao	fotv	fortv
Business Interface	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	or	fotv	fotv	ao	fotv	fotv	fotv	cfgostv	ao	fotv	fot	ao	fotv	fiotv
Business Object	no	nor	no	nor	no	nor	nor	nor	o	no	or	or	or	o	o	cgos	o	o	o	o	cgos	o	o	o	o	o
Business Process	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	o	fotv	fotv	ao	fotv	cfgortv	cfgortv	fotv	ao	cfgorstv	fot	ao	fotv	fortv
Product	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	or	fotv	fotv	acgor	fotv	fotv	fotv	fotv	acgor	fotv	cfgost	acgo	fotv	cfgortv
Representation	no	nor	no	nor	no	nor	nor	nor	o	no	or	or	or	o	o	or	o	o	o	o	or	o	o	cgos	o	o
Business Role	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	or	fotv	fotv	ao	fiotv	fiortv	fiortv	cfgotv	ao	fiortv	fot	ao	cfgostv	fiortv
Business Service	no	nor	no	nor	no	nor	nor	nor	no	no	or	or	o	fotv	fotv	ao	fotv	fotv	fotv	fotv	ao	fotv	fot	ao	fotv	cfgostv

Väzby Metamodel

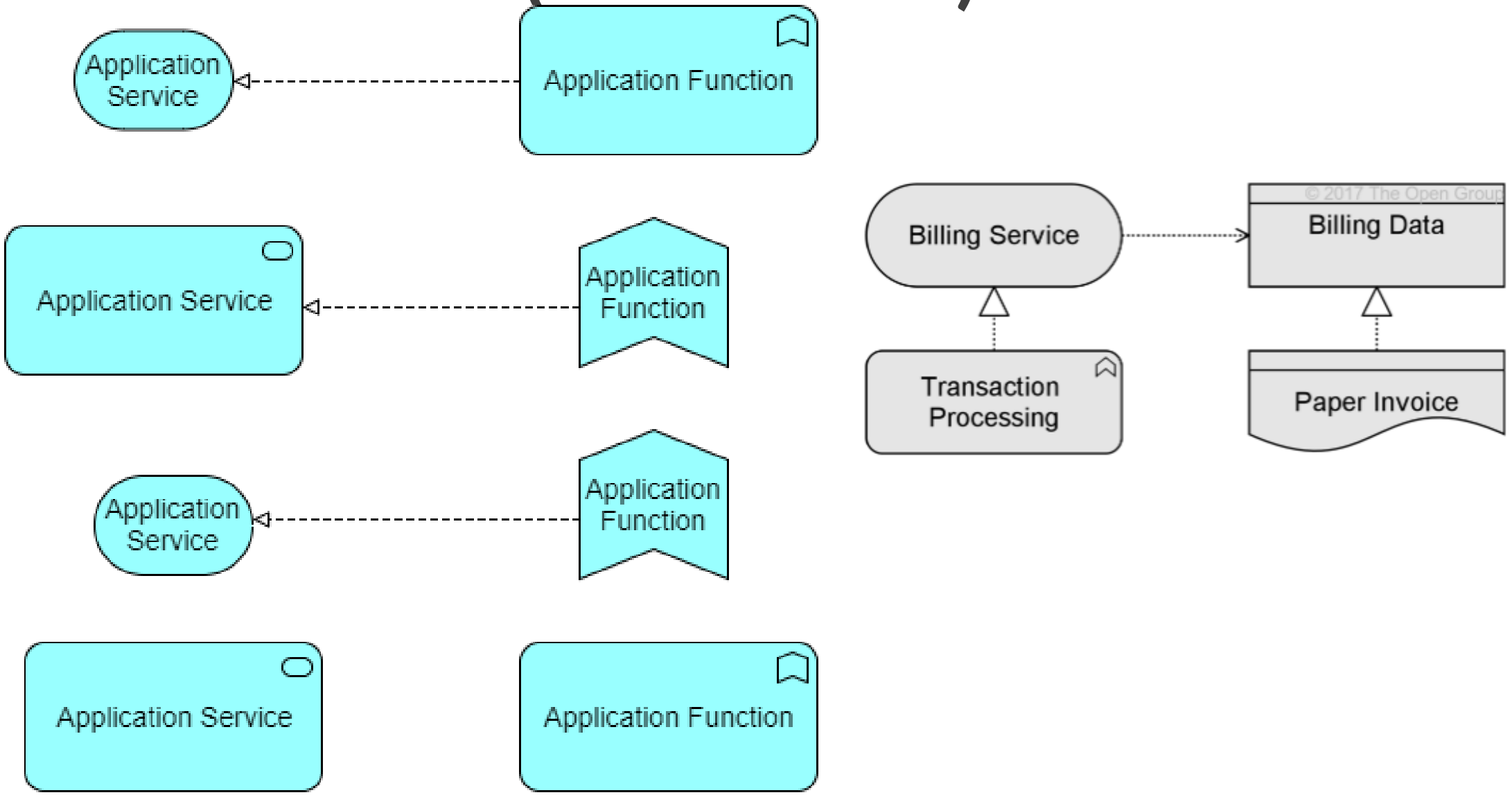


Väzba Realizácia (Realization)

- **Typ**: **Štrukturálna** (Structural)
- **Definícia**:
  - Táto väzba hovorí, že **entita vo smere šípky hraje kritickú rolu** pri **vytváraní**, **existencií**, **dosiahnutí cieľov** alebo **činností**, **viac abstraktnej entity**.
- **Príklad použitia**:
  - Aplikačná služba (Application Service) je realizovaná aplikačnou funkciou (Application Function). Biznis služba (Business Service) je realizovaná biznis procesom (Business Process) alebo biznis funkciou (Business function)

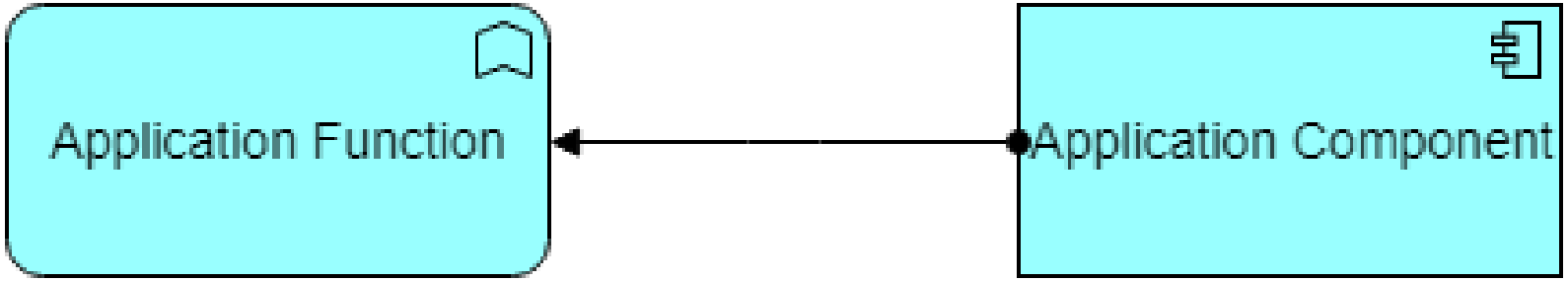


Väzba Realizácia (Realization)

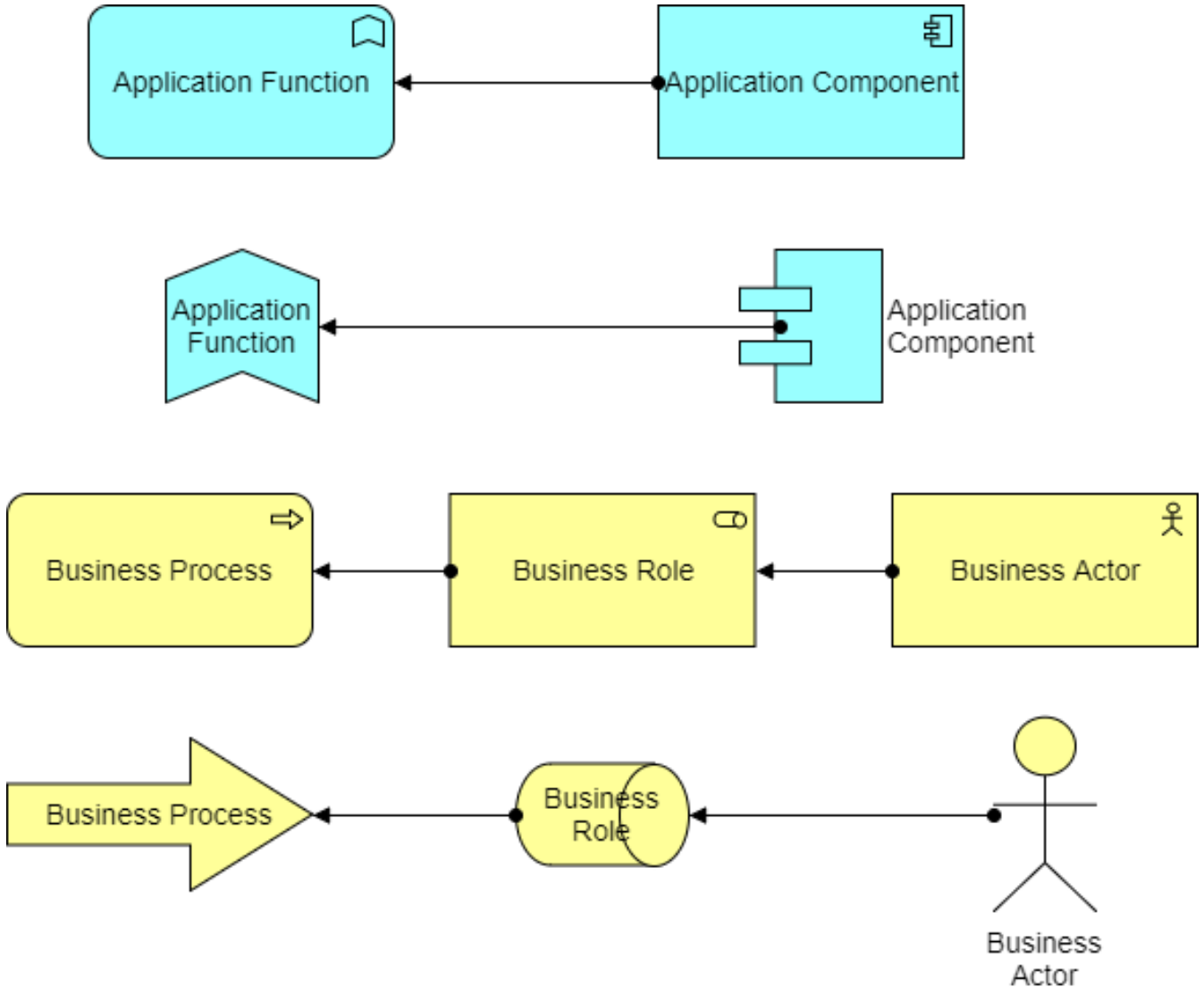


Väzba Priradenie (Assignment)

- **Typ: Štrukturálna** (Structural)
- **Definícia:**
  - Táto väzba priradzuje **určité chovanie** (Behavior) **aktívnemu elementu** (Active Structure). Väzba má od ArchiMate 3.0 **smerovosť** a je **ťahaná vždy** od **aktívneho elementu** k **elementu**, ktorý **reprezentuje chovanie**. V určitých prípadoch **môže byť** táto **väzba ťahaná** aj medzi **aktívnymi elementami**, napríklad od **aktéra** (Business Actor) **k biznis role** (Business role).
- **Príklad použitia:**
  - Aplikačná funkcia (Application Function), ktorá reprezentuje chovanie je priradená k aplikačnej komponente, ktorá je aktívnym elementom.

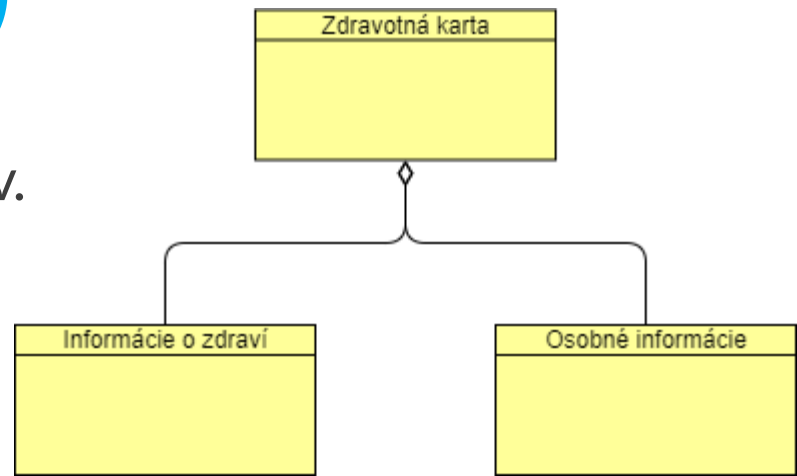


Väzba Priradenie (Assignment)

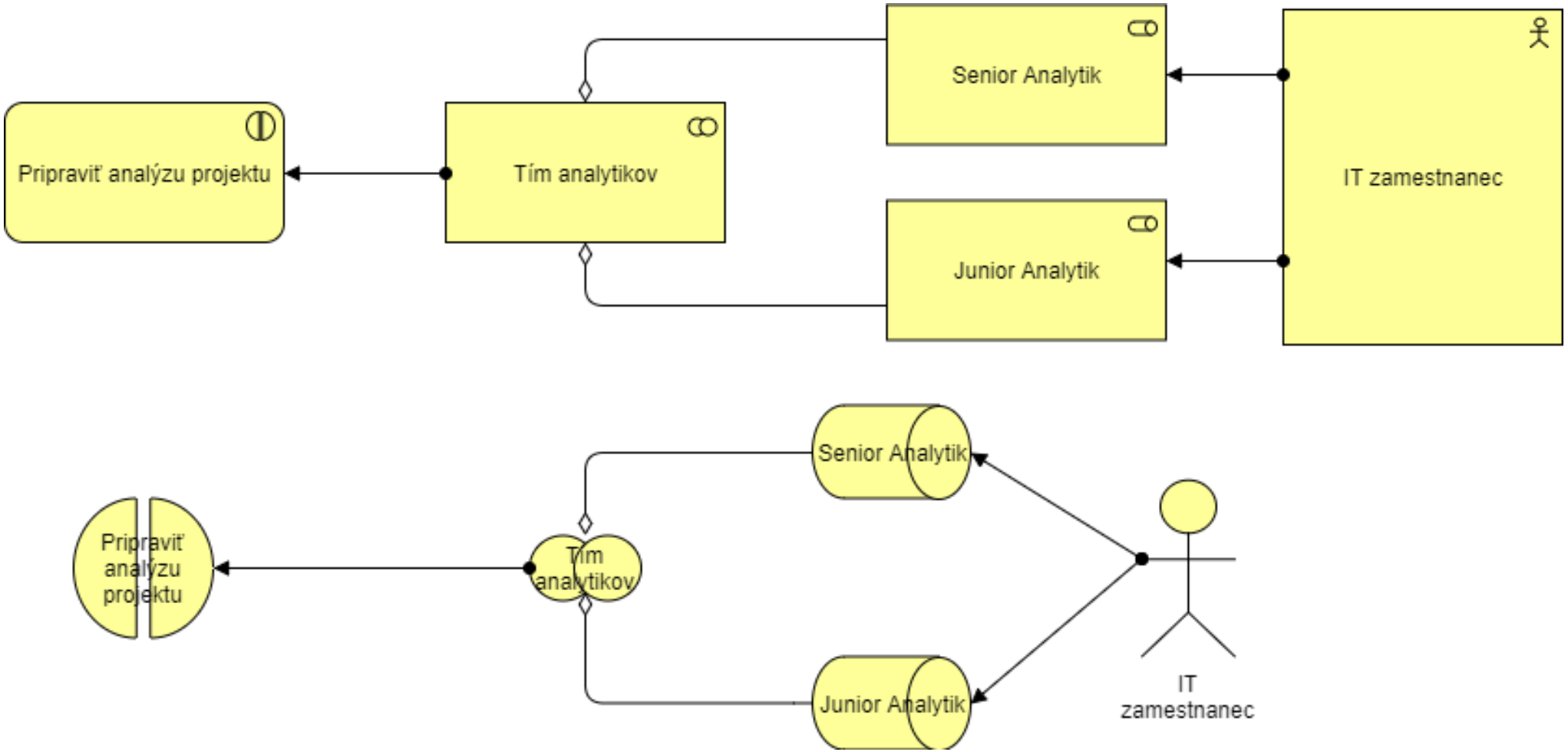


Väzba Agregácia (Aggregation)

- **Typ**: **Štrukturálna** (Structural)
- **Definícia**:
  - Táto väzba definuje, že **element** k **nemu** je **agregácia vedená** (na ktorého **strane sa nachádza diamant**) **pozostáva** z **viacerých iných elementov**. A teda, že **element** od **ktorého** je **agregácia vedená** je **súčasťou nejakého** celku. Agregácia **je vždy povolená medzi elementami rovnaké typu**. ArchiMate ďalej definuje medzi akými typmi elementov je táto väzba povolená.
  - Na rozdiel od kompozície **môže byť 1 element súčasťou viac celkov (agregácii)**
- **Príklad použitia**:
  - Priradenie rolí do kolaborácii, skladba dátových objektov.



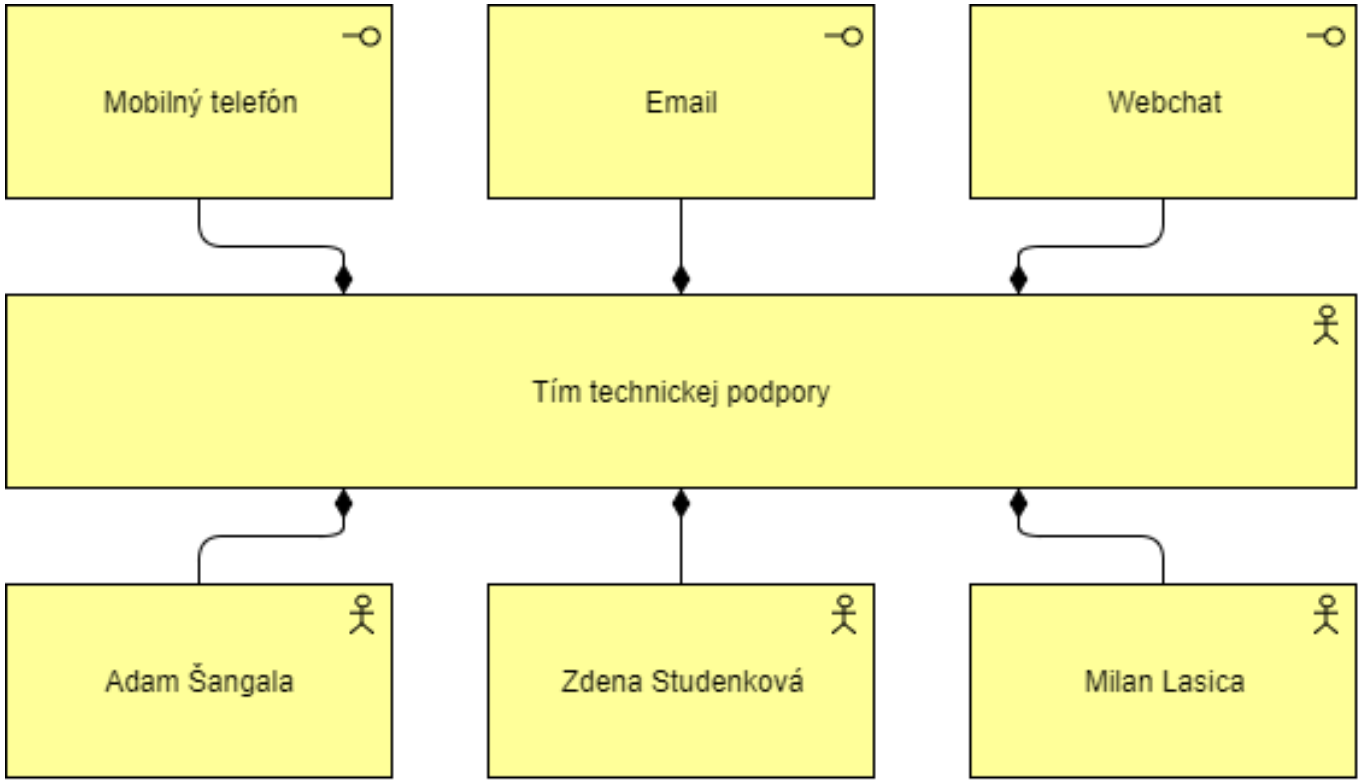
Väzba Agregácia (Aggregation)

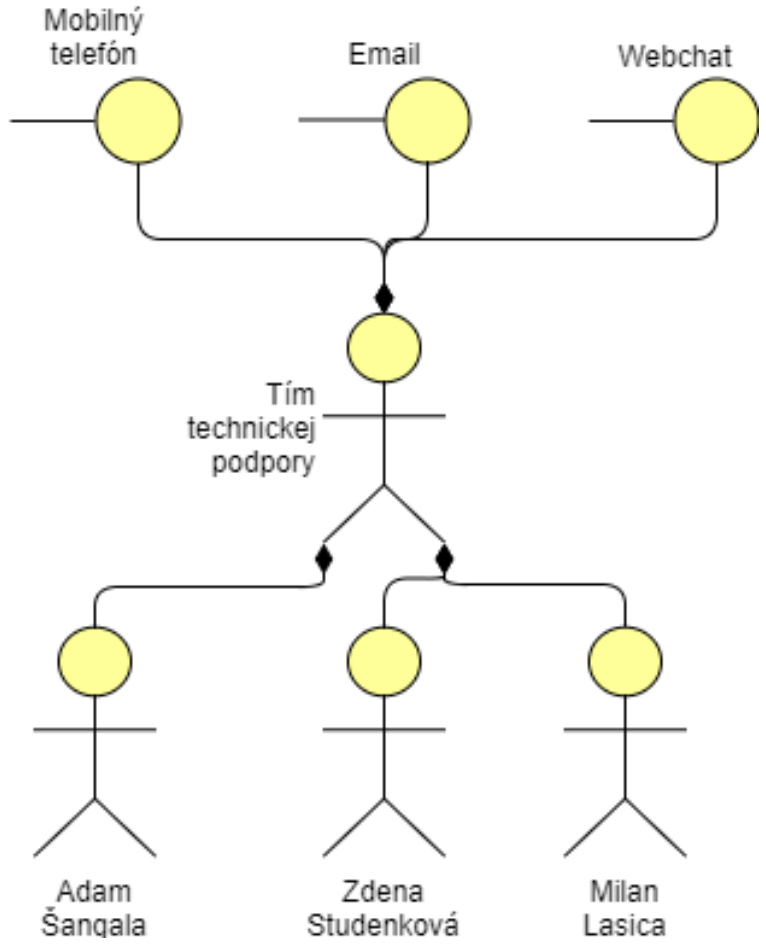


Väzba Kompozícia (Composition)

- **Typ**: **Štrukturálna** (Structural)
- **Definícia**:
  - Táto väzba definuje **vnútorné štruktúry** (**súdržnosť**) vo **vnútri architektúry**. **Element**, k **nemu je kompozícia vedená** (**na** ktorého **strane** sa nachádza **diamant**) **pozostáva z viac iných elementov**. Element od, ktorého je agregácia vedená je súčasťou nejakého celku.
  - **Kompozícia** je **vždy povolená** medzi **elementami rovnaké typu**. ArchiMate ďalej definuje medzi akými typmi elementov je táto väzba dovolená.
- **Príklad použitia**:
  - Rozpad komponent aplikácie do subkomponent. Priradenie interface ostatným aktívnym objektom. Priradenie aktérov do tímu a pod.

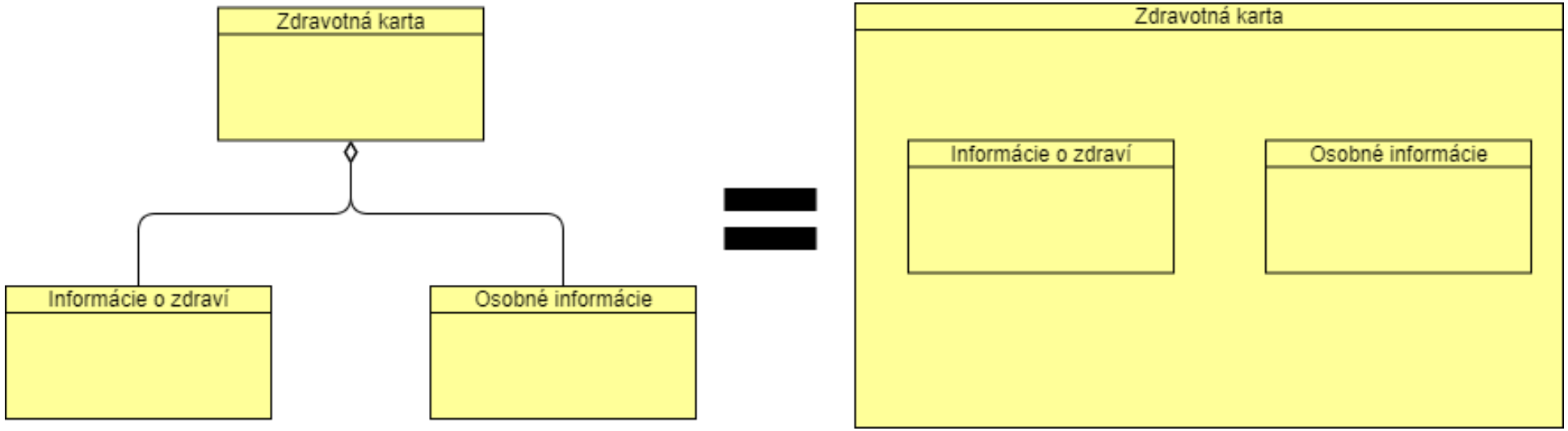
Väzba Kompozícia (Composition)





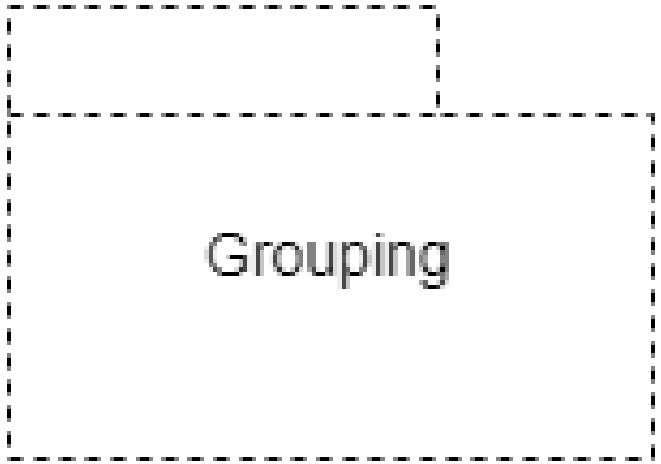
Vnorenie (nesting)

- Existuje ešte ďalšia možnosť ako vyjadriť, že niektoré elementy sú **navzájom vo vzťahu kompozície** (Composition) alebo **agregácie** (Aggregation). Nasledujúce zápisy sú, čo sa týka významu zhodné.

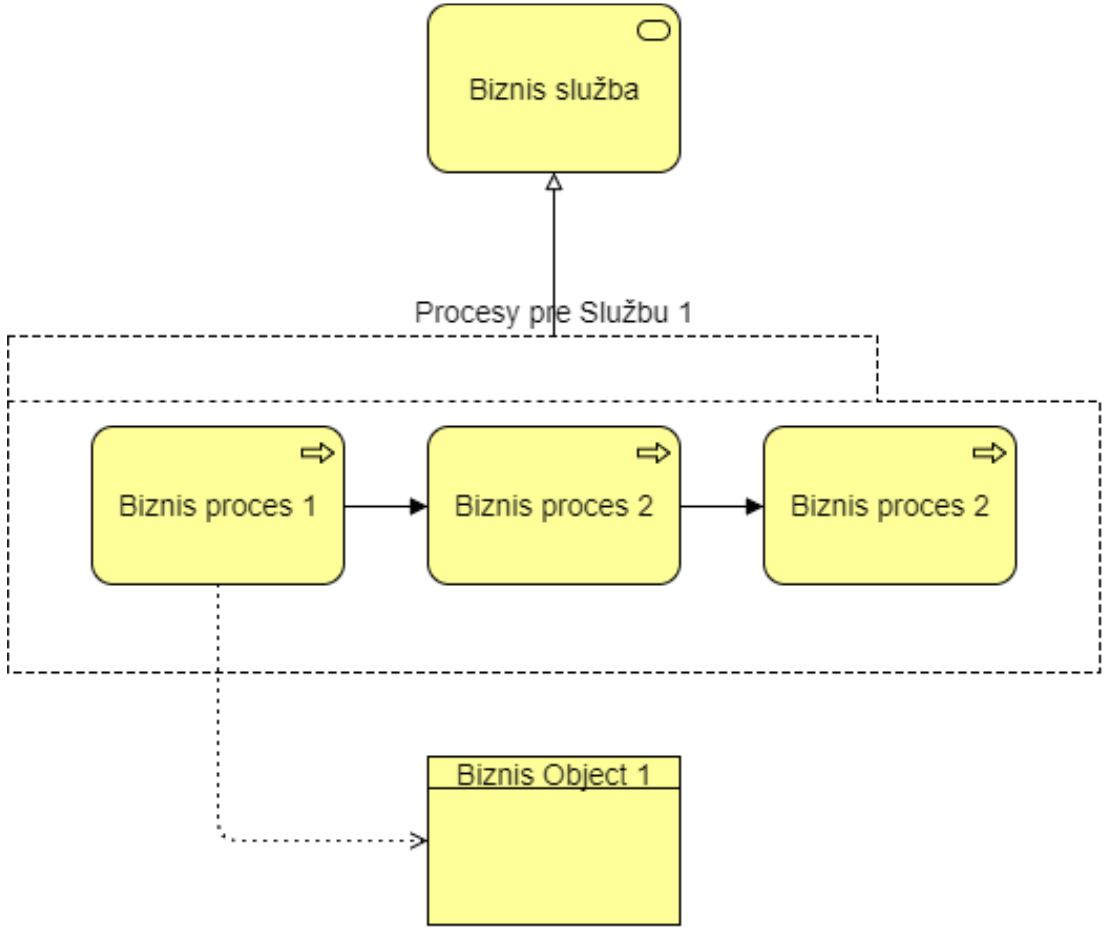


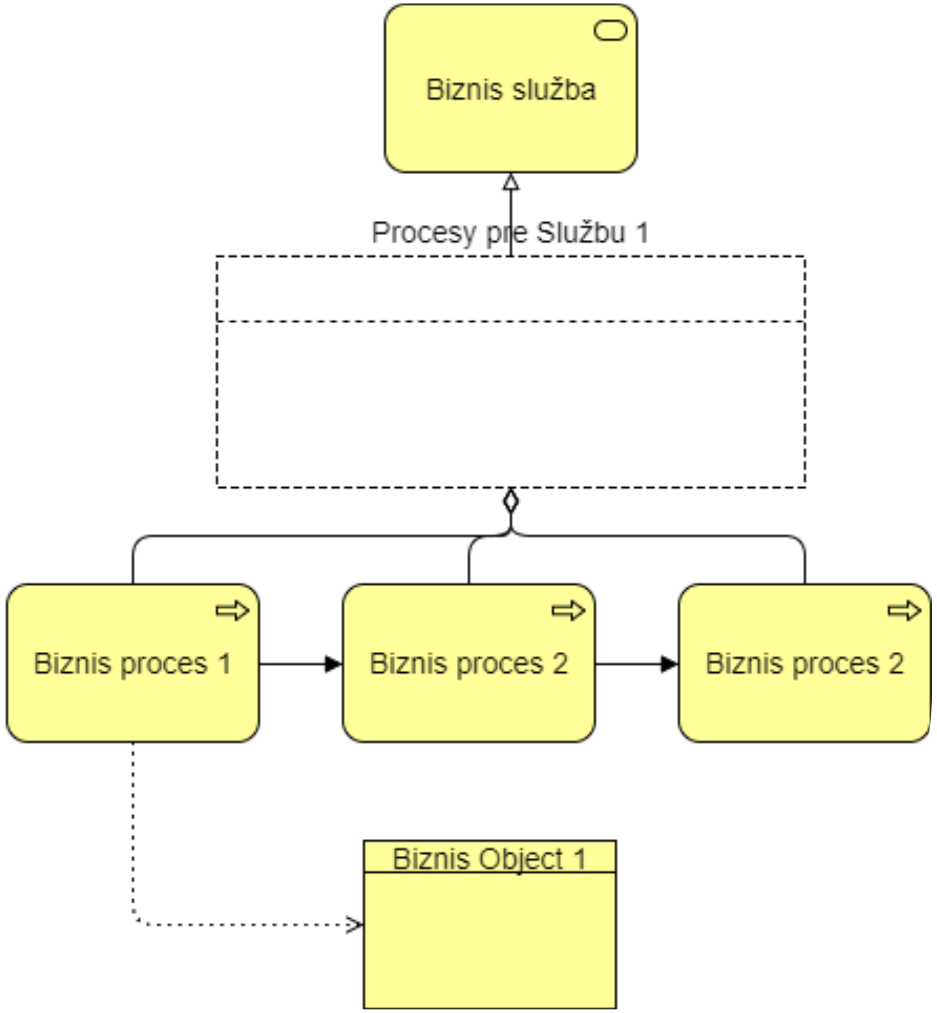
Zoskupovanie (Grouping)

- Grouping **element agreguje** (Aggregate alebo Composite) **koncepty** (**elementy** alebo **väzby**), ktoré k sebe náležia na **základe nejakej spoločnej/zdieľanej charakteristiky**.
- Koncepty **môžu patriť do viac ako 1 skupiny**.
- Existujú **2 spôsoby zápisu**.

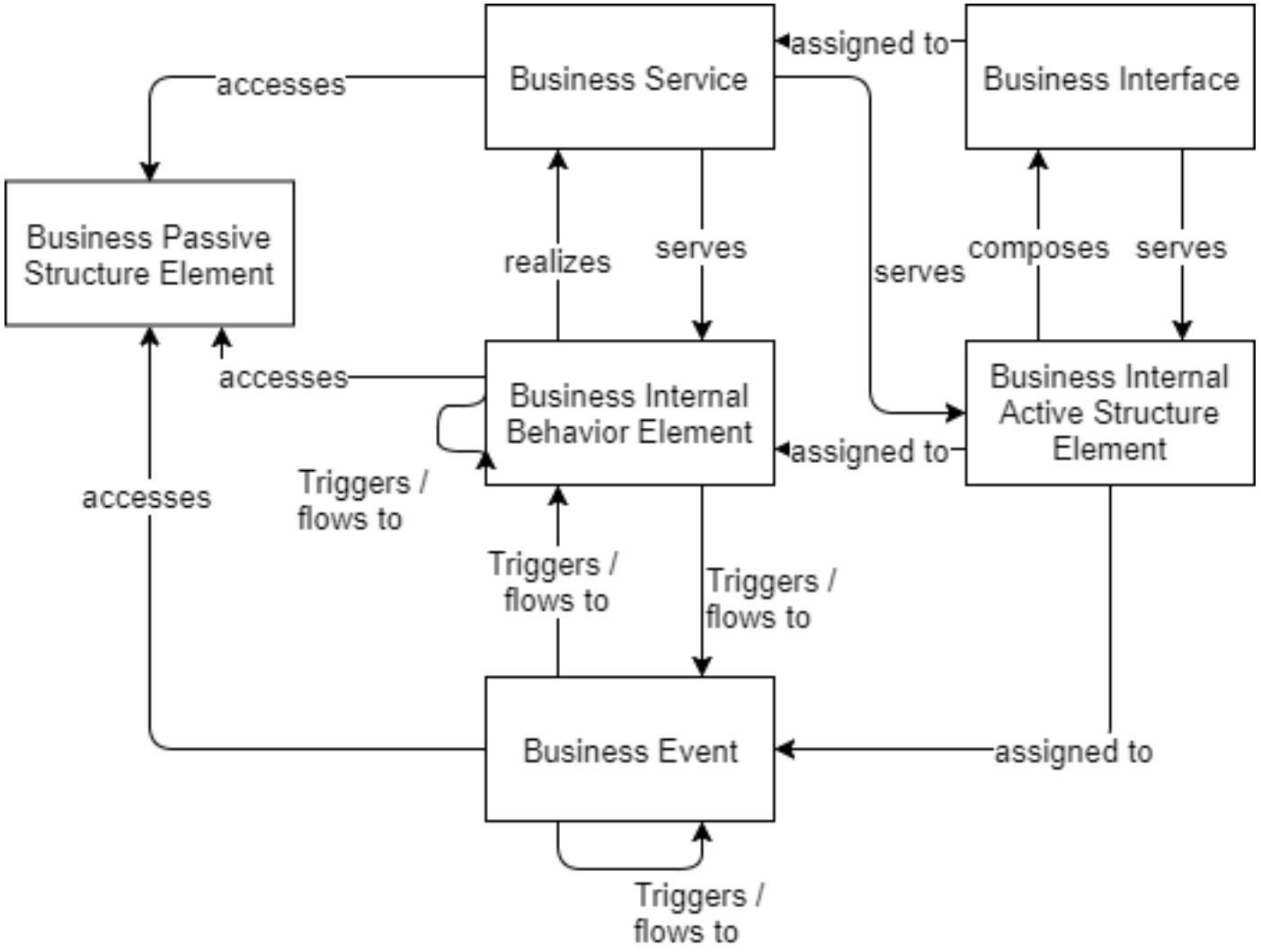


Zoskupovanie (Grouping)

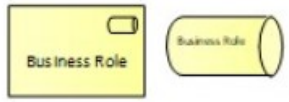
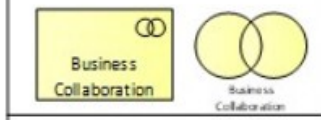
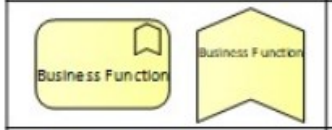
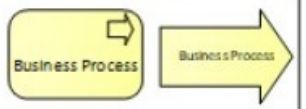




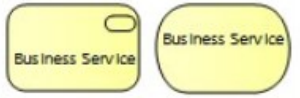
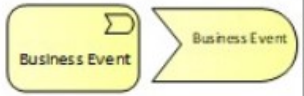
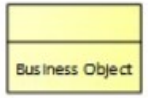
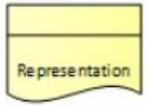
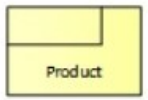
Biznis vrstva - Metamodel



Biznis vrstva - Elementy 1

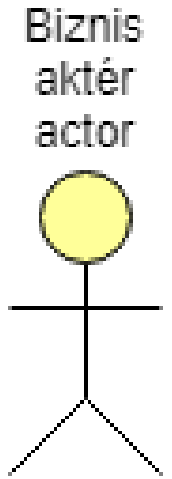
Element	Definice	Aspekt
	A business actor is a business entity that is capable of performing behavior.	Active structure
	A business role is the responsibility for performing specific behavior, to which an actor can be assigned, or the part an actor plays in a particular action or event.	Active structure
	A business collaboration is an aggregate of two or more business internal active structure elements that work together to perform collective behavior.	Active structure
	A business interface is a point of access where a business service is made available to the environment.	Active structure
	A business function is a collection of business behavior based on a chosen set of criteria (typically required business resources and/or competences), closely aligned to an organization, but not necessarily explicitly governed by the organization.	Behavior
	A business process represents a sequence of business behaviors that achieves a specific outcome such as a defined set of products or business services.	Behavior
	A business interaction is a unit of collective business behavior performed by (a collaboration of) two or more business roles.	Behavior

Biznis vrstva - Elementy 2

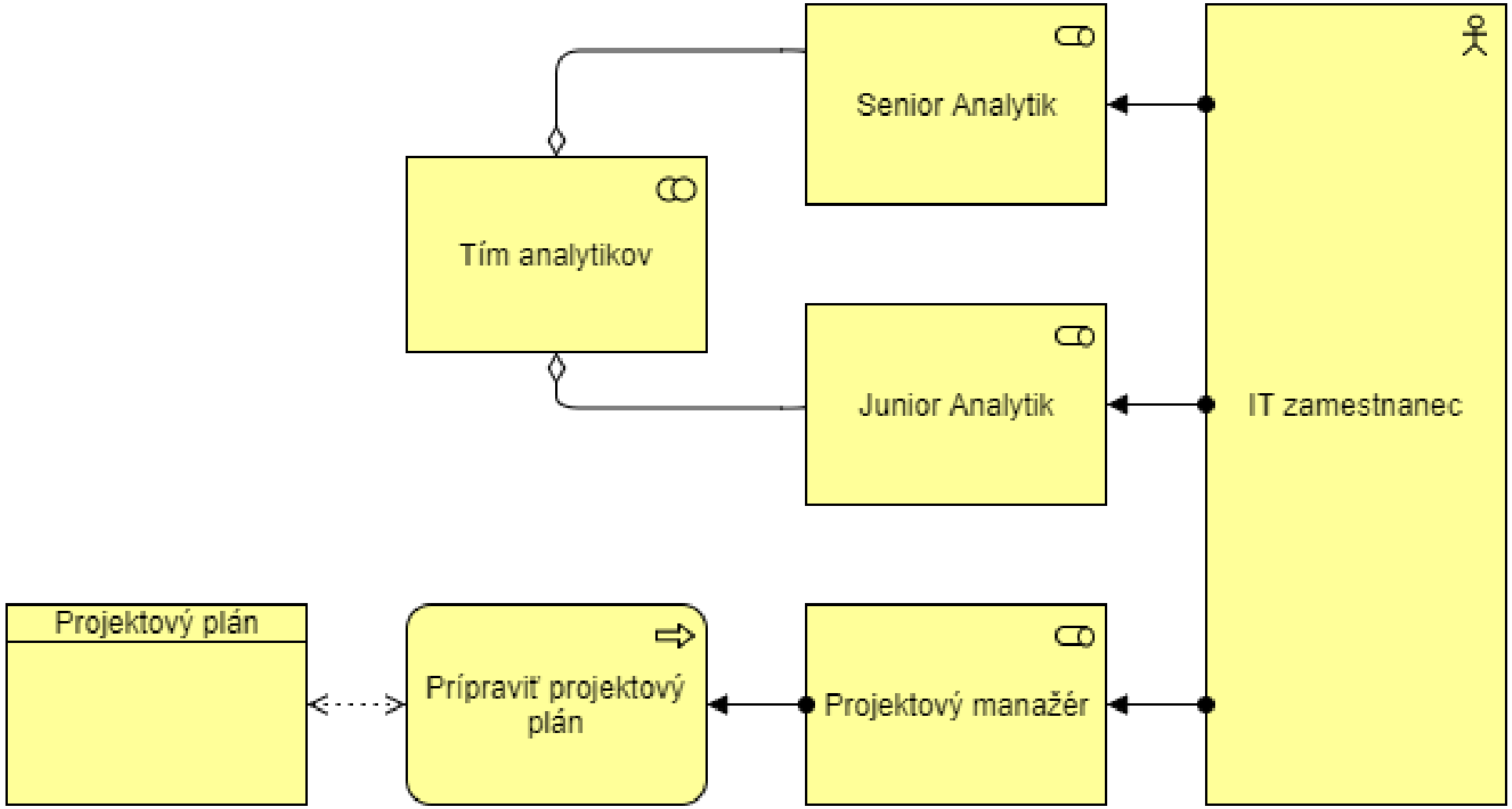
Element	Definice	Aspekt
	A business service represents an explicitly defined exposed business behavior.	Behavior
	A business event is a business behavior element that denotes an organizational state change. It may originate from and be resolved inside or outside the organization.	Behavior
	A business object represents a concept used within a particular business domain.	Passive Structure
	A representation represents a perceptible form of the information carried by a business object.	Passive Structure
	A contract represents a formal or informal specification of an agreement between a provider and a consumer that specifies the rights and obligations associated with a product and establishes functional and non-functional parameters for interaction.	Passive Structure
	A product represents a coherent collection of services and/or passive structure elements, accompanied by a contract/set of agreements, which is offered as a whole to (internal or external) customers. *Poznámka: tento element je často uváděn jako součást Business Layer, nicméně podle ArchiMate superstruktury patří do kompozitních elementů, podobně jako location nebo grouping. Proto je uveden v obou přehledech.*	Passive Structure **Composite**

Element Aktér v biznis vrstve (Actor)

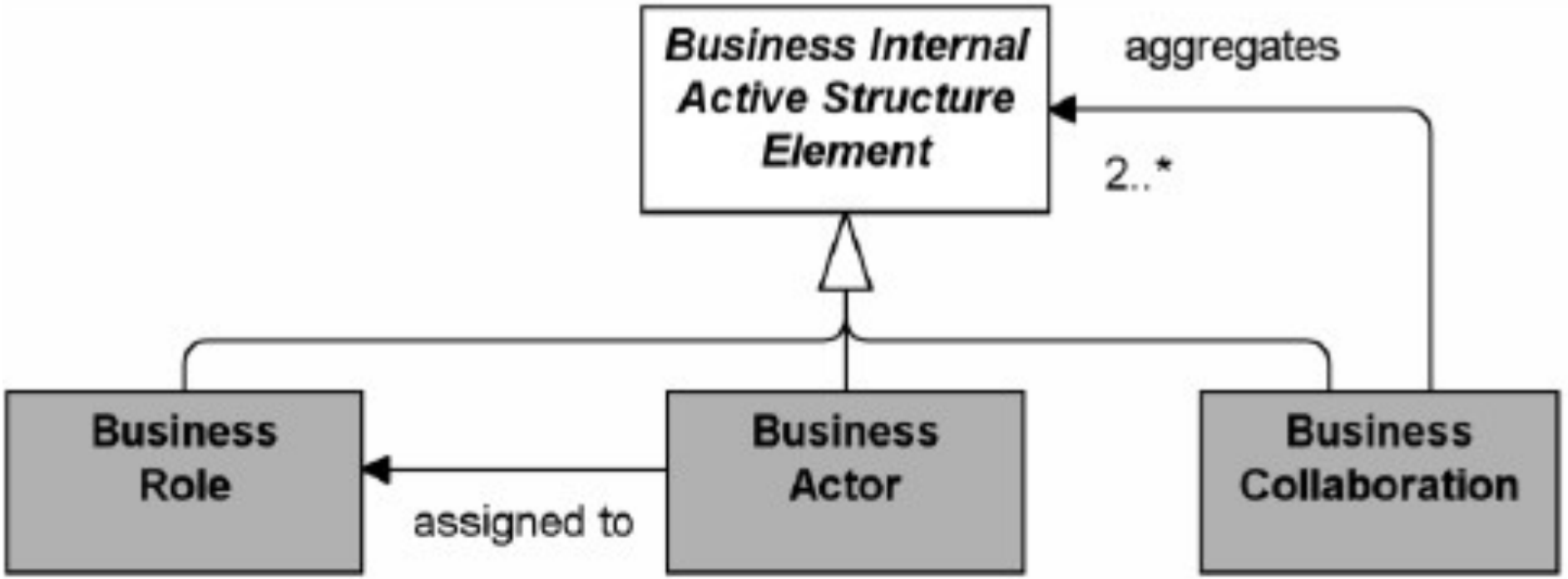
- **Typ**: **aktívny** (Structural Active)
- **Definícia**:
  - Biznis aktér (Business Actor) je entita, ktorá je schopná určitého chovania.
  - Aktéri môžu byť interní (zamestnanci) alebo externí – zákazníci, dodávatelia, partneri.
  - Aktéri môžu reprezentovať konkrétneho človeka, oddelenie alebo organizáciu a to na rôznych úrovniach abstrakcie.
- **Príklad použitia**:
  - Pracovník oddelenia RISK
  - Oddelenie RISK
  - Adam Šangala
  - Dodávateľ (generický dodávateľ)
  - ABC s.r.o. (konkrétny dodávateľ)



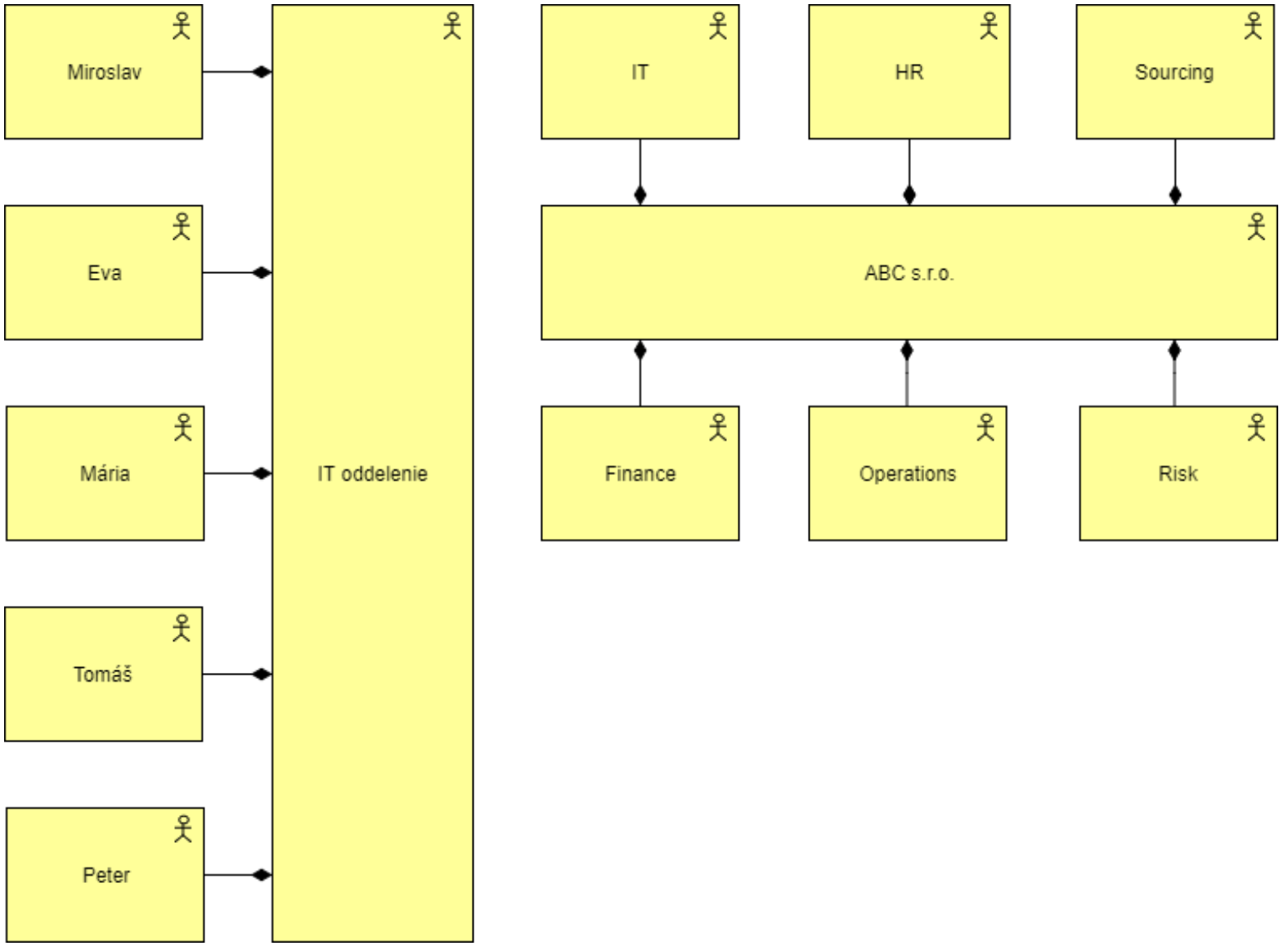
Element Aktér v biznis vrstve (Actor)



Element Aktér v biznis vrstve (Actor)

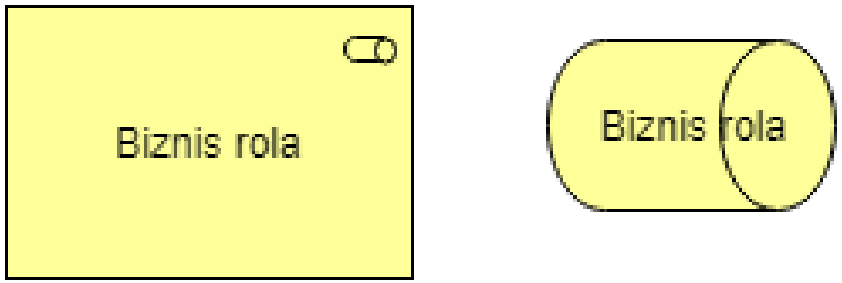


Element Aktér v biznis vrstve (Actor)

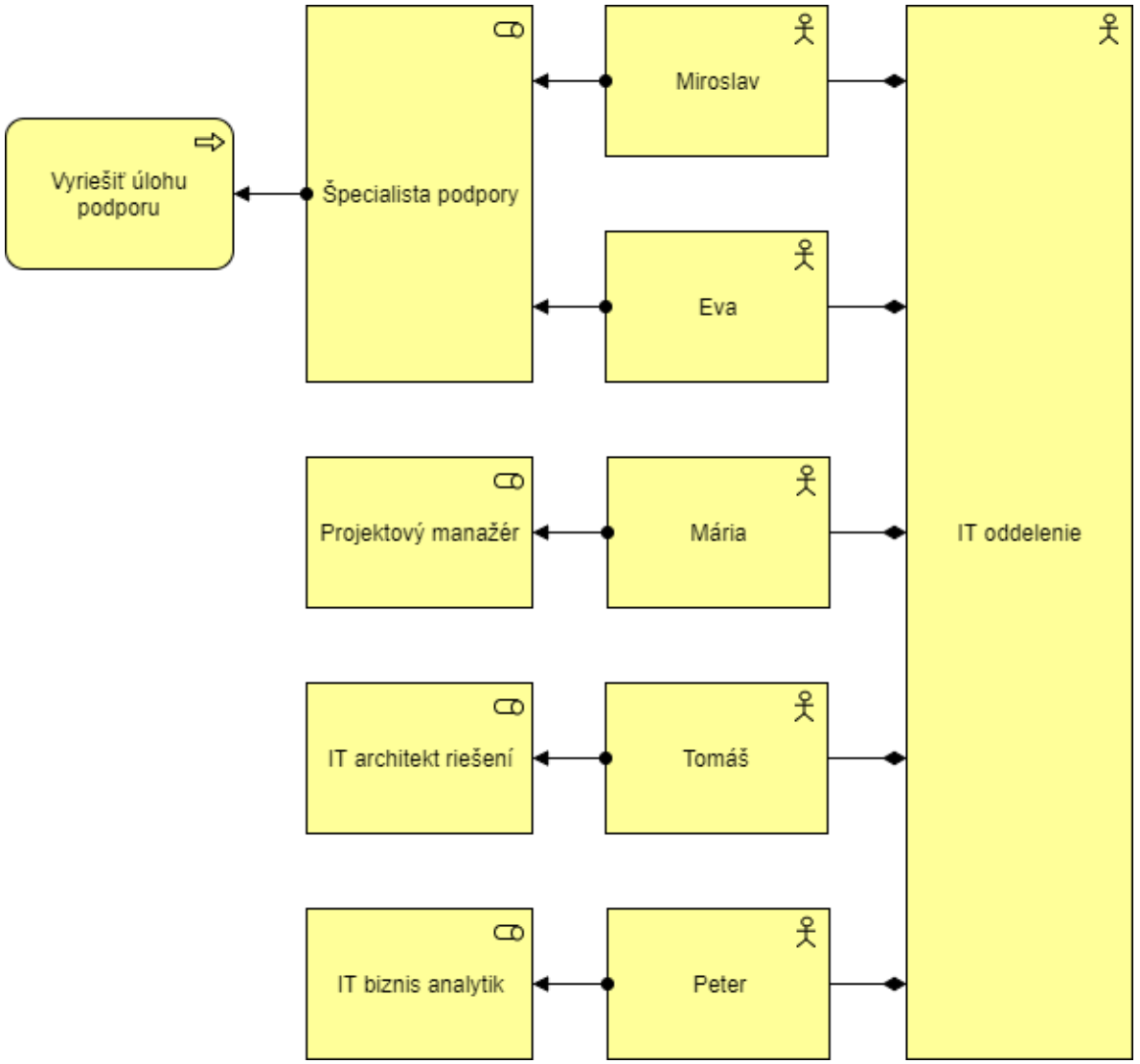


Element Rola v biznis vrstve (Role)

- **Typ**: **aktívny** (Structural Active)
- **Definícia**:
  - Predstavuje zodpovednosť za špecifické chovanie, ku ktorému môže byť aktér (Business Actor) priradený alebo za úlohu, ktorú aktér hraje v akcii či udalosti.
  - Rola predstavuje zodpovednosť alebo schopnosti, ktoré sú priradené určitému biznis procesu (Business Process) alebo biznis funkcii (Business Function). Aktér (Business Actor), ktorý je priradený k tejto roli je zodpovedný za to, že zodpovedajúce chovanie (Biznis proces či biznis funkcia) je vykonaná a to buď, že ho aktér vykoná priamo, deleguje ho či vykonanie riadi (manage)
- **Príklad použitia**:
  - Projektový manažér
  - Výkonný riaditeľ
  - Projektová podpora
  - Autorita na zmena
  - Senior špecialista
  - IT Analytik

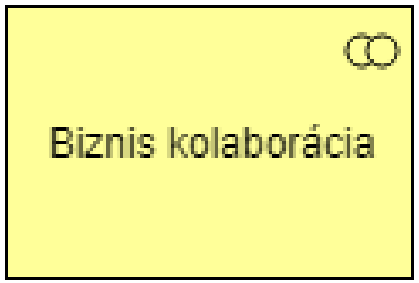


Element Rola v biznis vrstve (Role)

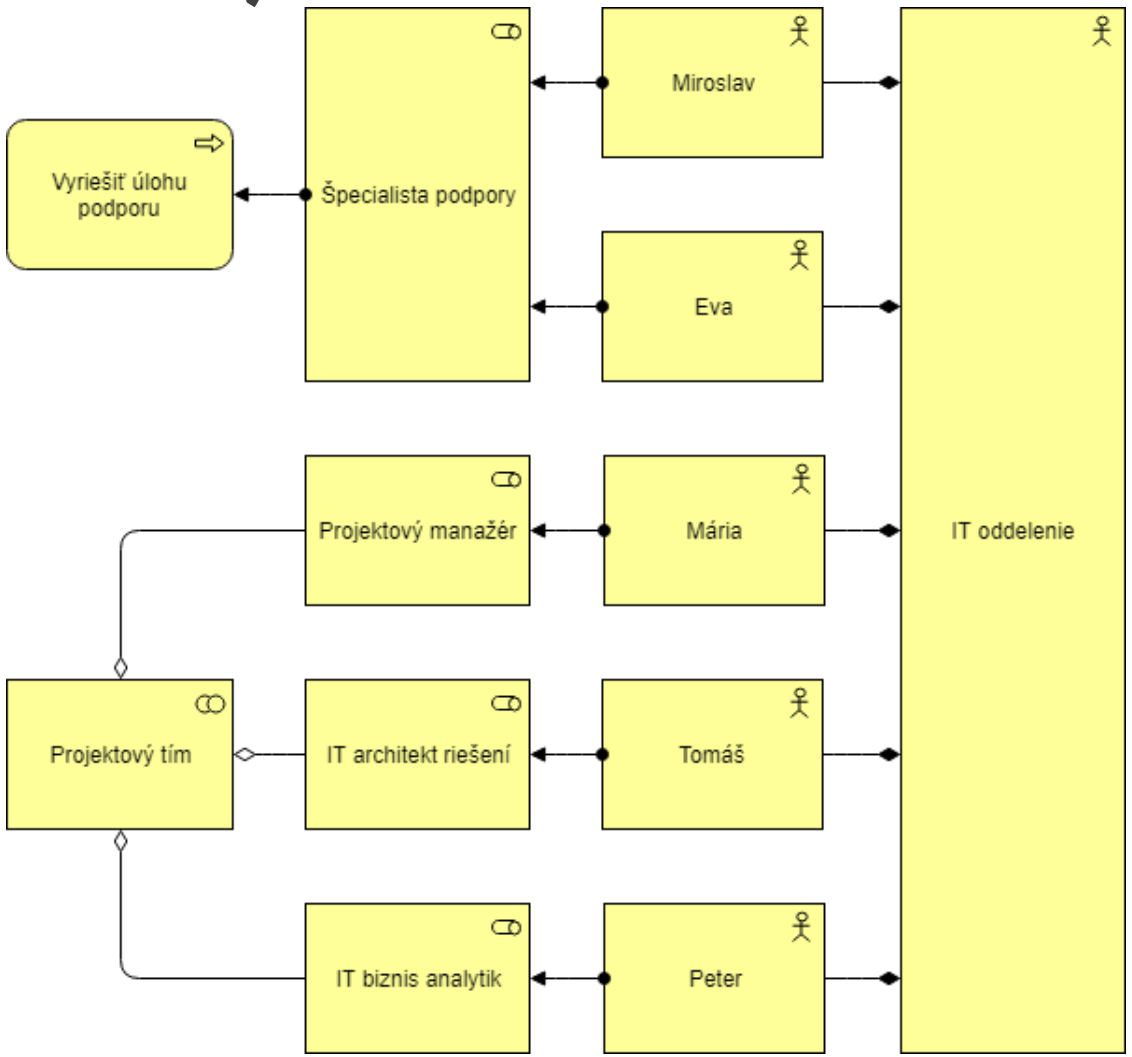


Element kolaborácia v biznis vrstve (Collaboration)

- **Typ**: **aktívny** (Structural Active)
- **Definícia**:
  - Agreguje 2 alebo viac biznis aktívnych štruktúr (Active Structure – aktérov alebo role), ktoré musia spolupracovať, aby dosiahli spoločného účelu (spoločného chovania - behavior).
- **Príklad použitia**:
  - Projektový tým
  - Komisia pre schválenie rozpočtu stavby
  - Projektová rada

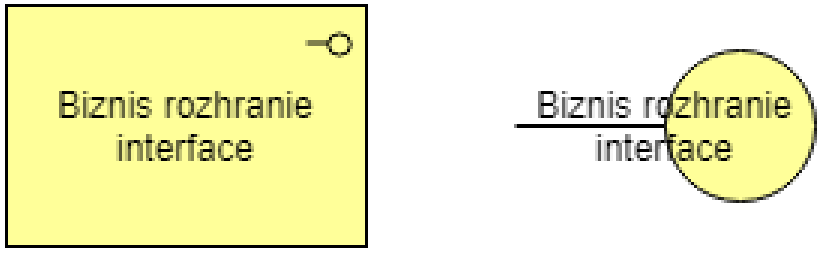


Element kolaborácia v biznis vrstve (Collaboration)

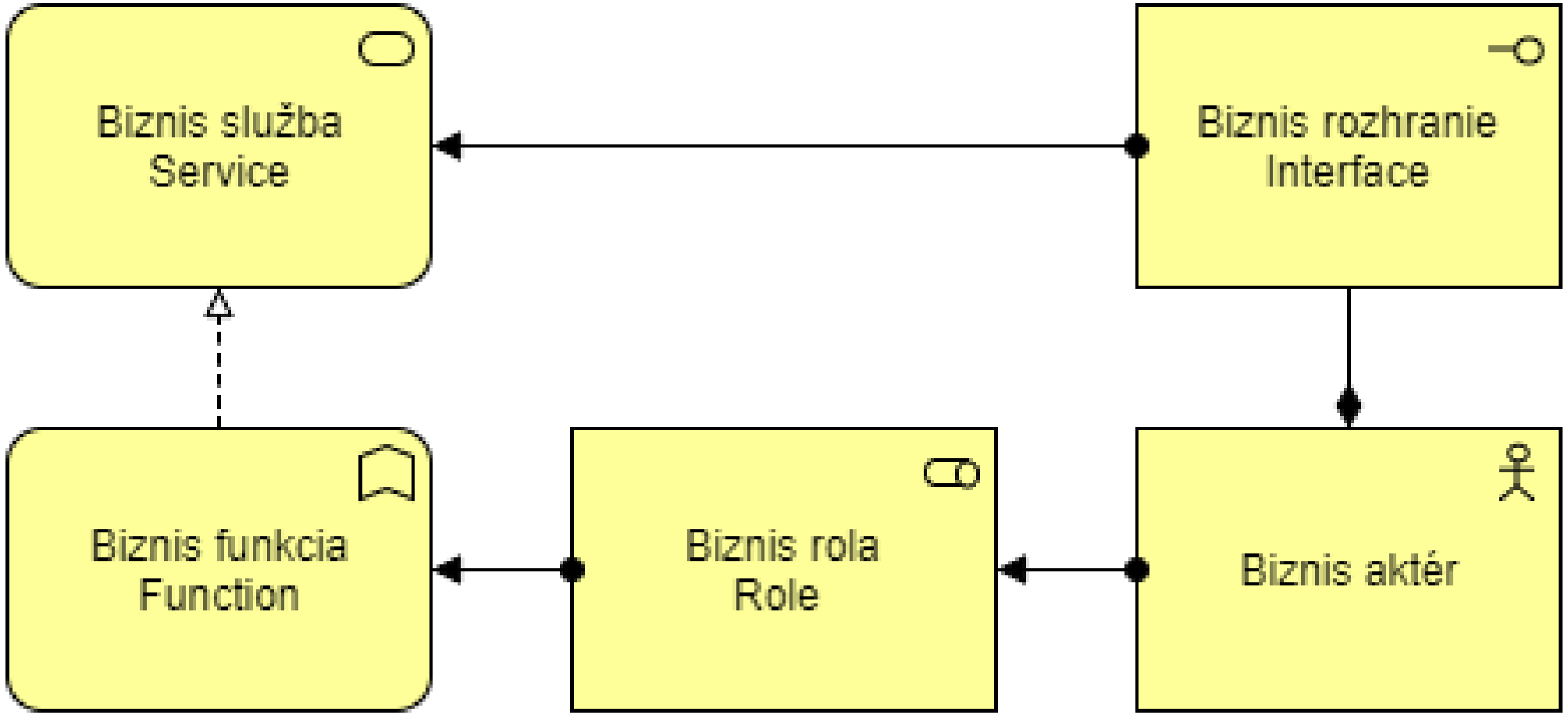


Element Rozhranie v biznis vrstve (Interface)

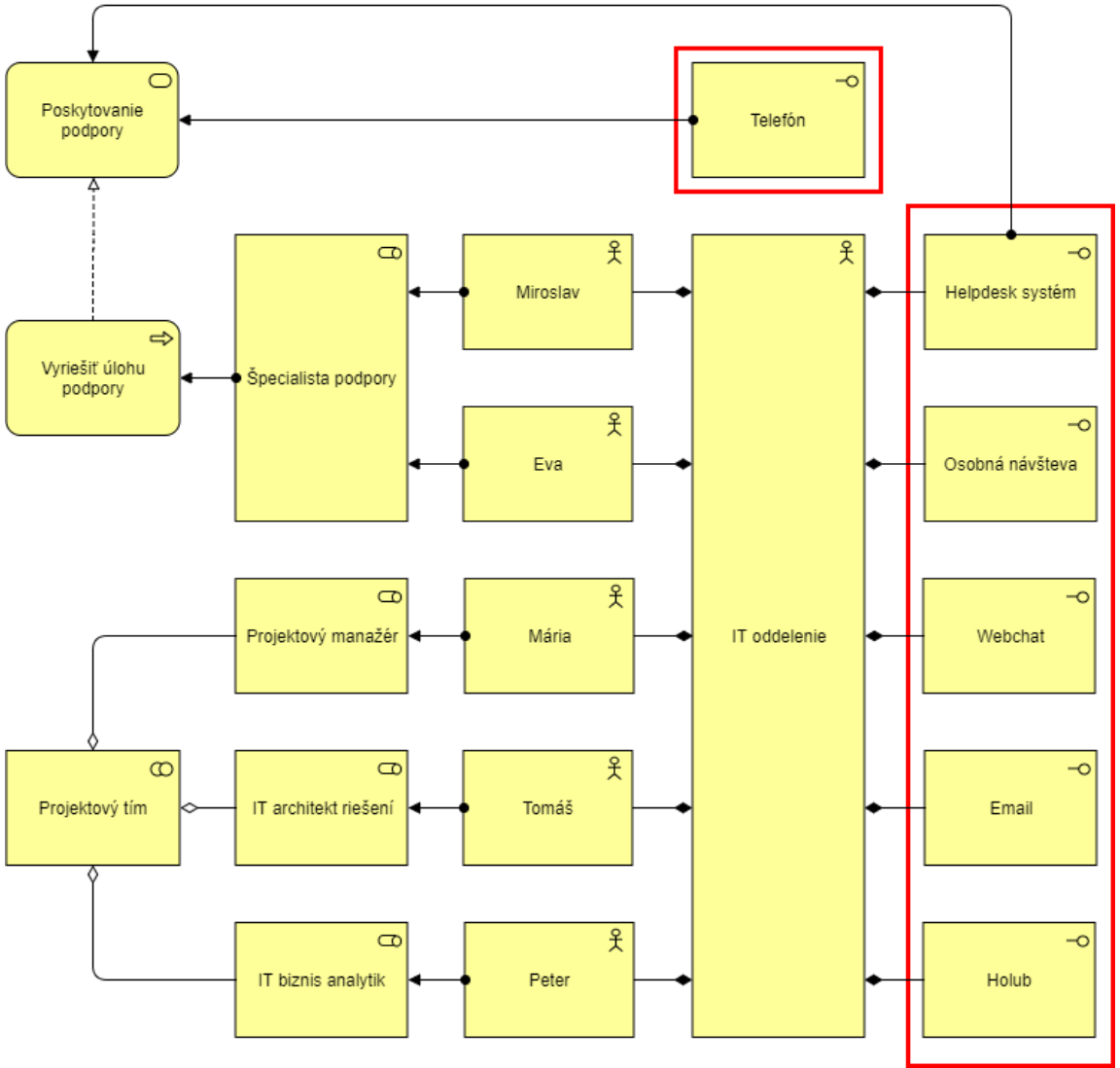
- **Typ**: **aktívny** (Structural Active)
- **Definícia**:
  - Je prístupový bod, kde je biznis služba dostupná z okolitého prostredia.
- **Príklad použitia**:
  - Telefón
  - Email
  - Webchat
  - Fyzické prítomnosť na pobočke



Element Rozhranie v biznis vrstve (Interface)

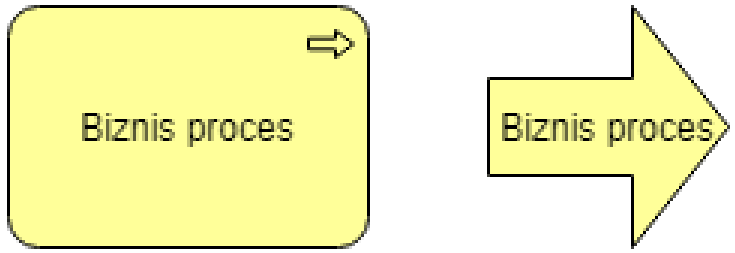


Element Rozhranie v biznis vrstve (Interface)

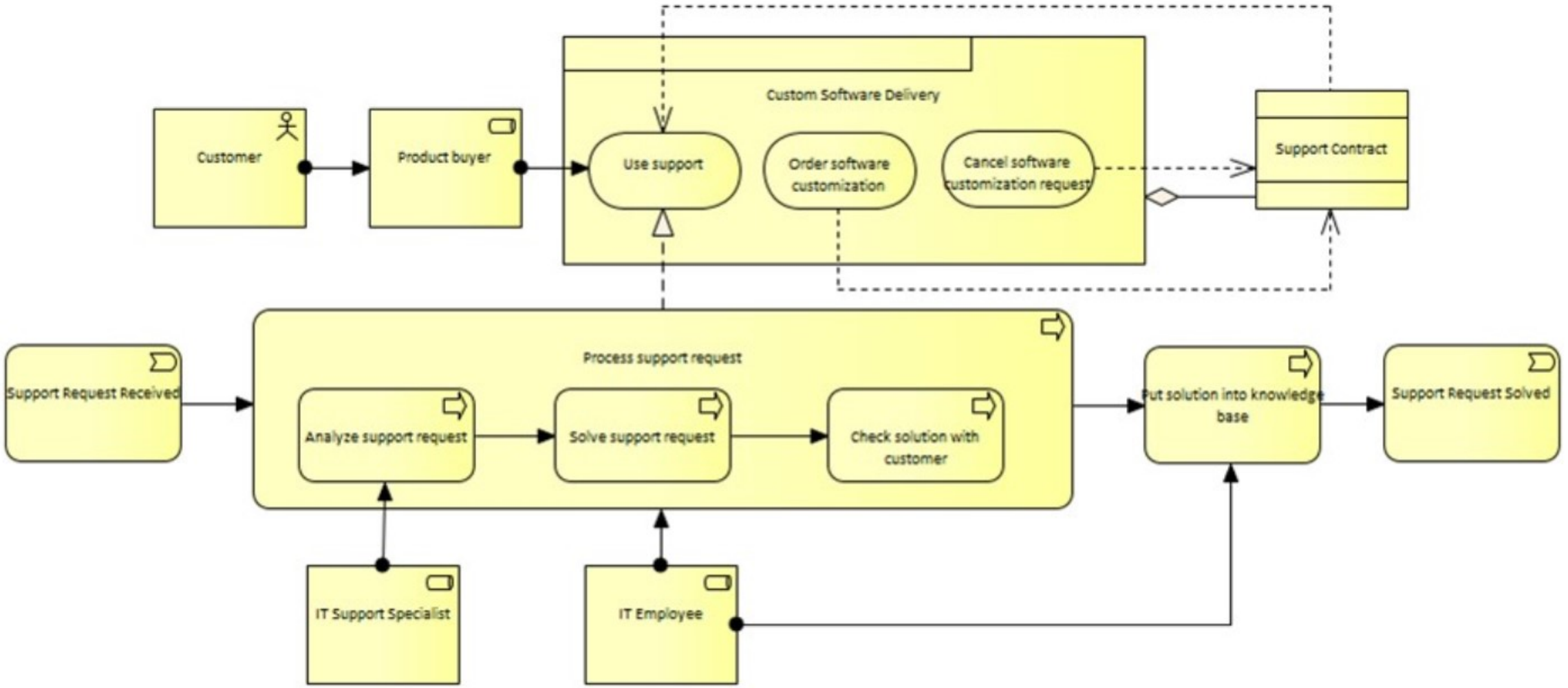


Element Proces v biznis vrstve (Process)

- **Typ**: **aktívny interný** (Internal Behavior)
- **Definícia**:
  - Reprezentuje sekvenciu akcií (či chovaní - Behavior), ktoré musia byť splnené pre dosiahnutie požadovaného výstupu, ktorý je definovaný napríklad jednými či viac produktami (Business Product) alebo biznis službami (Business Service)
- **Príklad použitia**:
  - Poslať faktúru
  - Prijať objednávku
  - Skontrolovať riešenie so zákazníkom

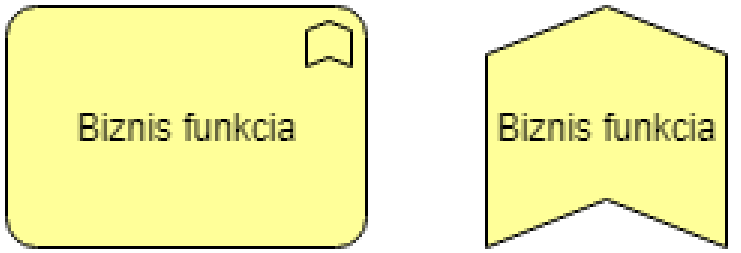


Element Proces v biznis vrstve (Process)

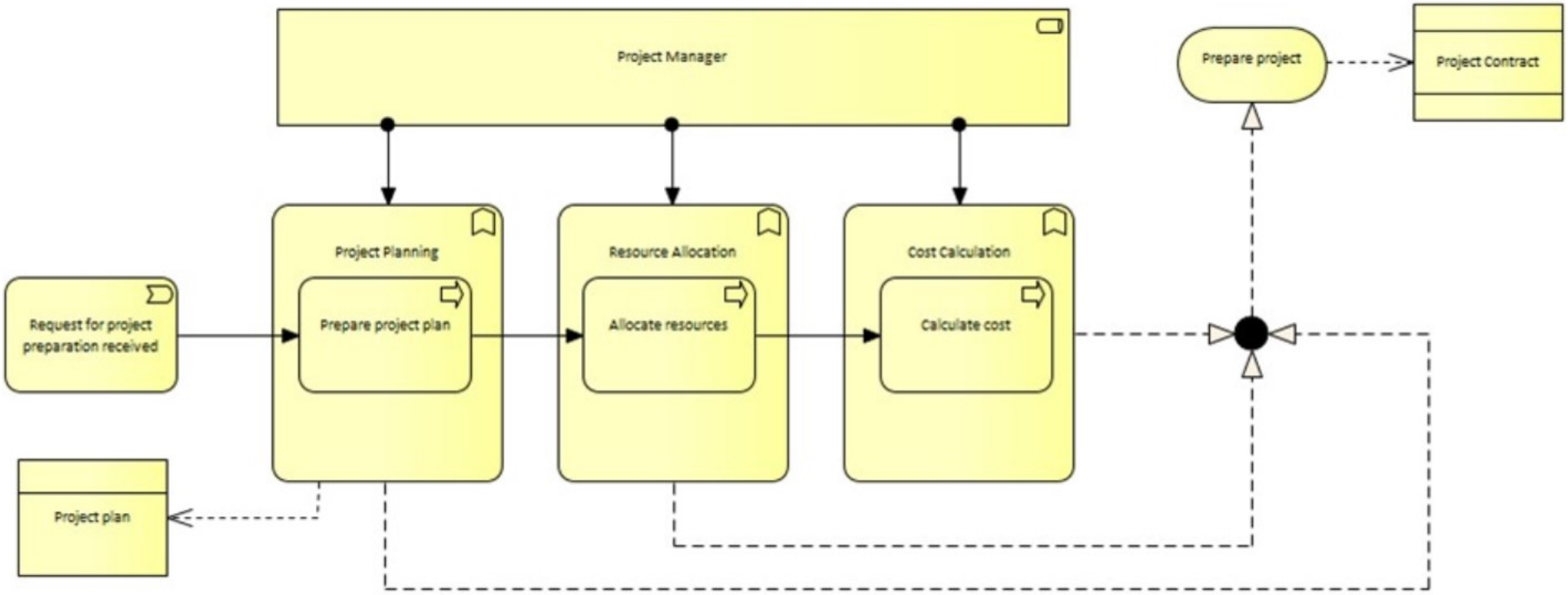


Element Funkcia v biznis vrstve (Function)

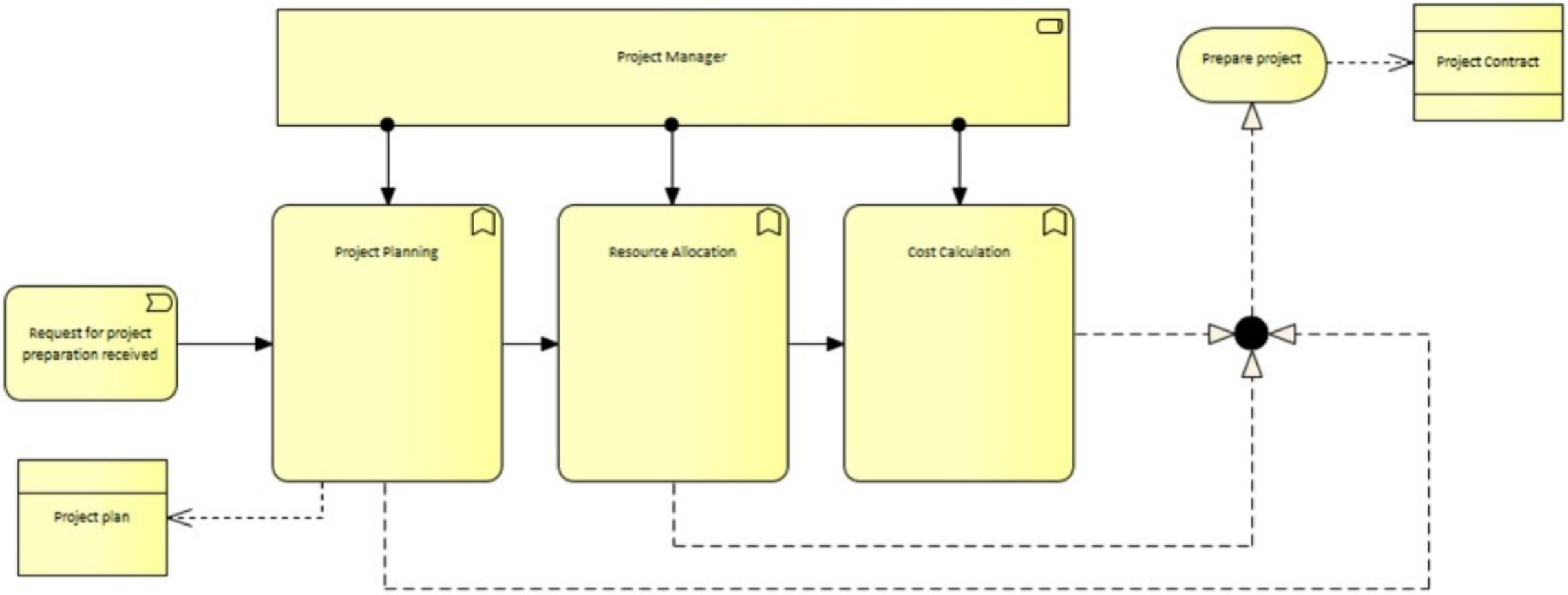
- **Typ**: **aktívny interný** (Internal Behavior)
- **Definícia**:
  - Predstavuje súbor chovaní, ktorý je definovaný na základe zvolenej množiny kritérií napríklad zdroje, biznis oblasť alebo kompetencie.
- **Príklad použitia**:
  - Projektové plánovanie
  - Alokovanie zdrojov
  - Výpočty nákladov



Element Funkcia v biznis vrstve (Function)

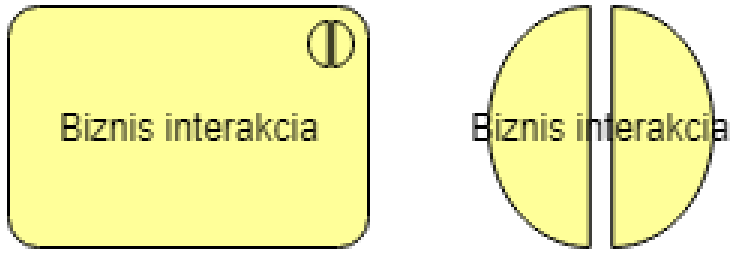


Element Funkcia v biznis vrstve (Function)

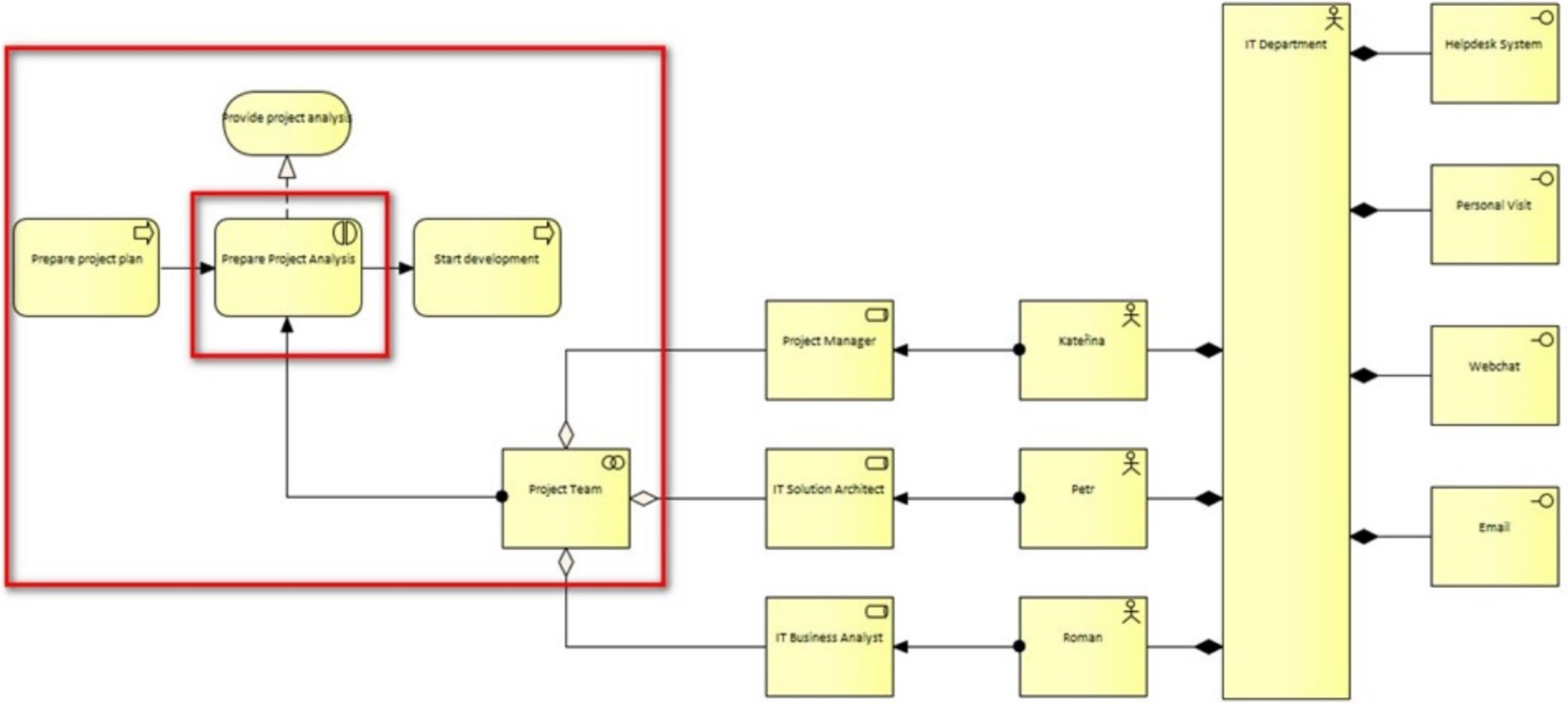


Element Interakcia v biznis vrstve (Interaction)

- **Typ**: **aktívny interný** (Internal Behavior)
- **Definícia**:
  - Predstavuje spoločné chovanie, ktoré je vykonávané jednou či viac biznis rolami spojených do kolaborácie (Business Collaboration)
- **Príklad použitia**:
  - Pripraviť analýzu projektu

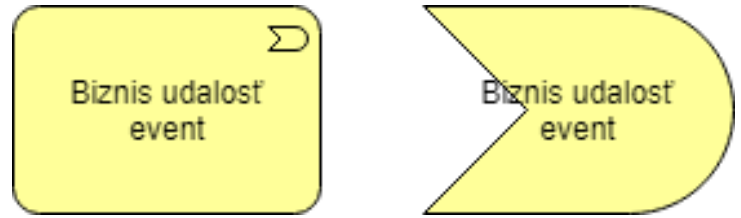


Element Interakcia v biznis vrstve (Interaction)

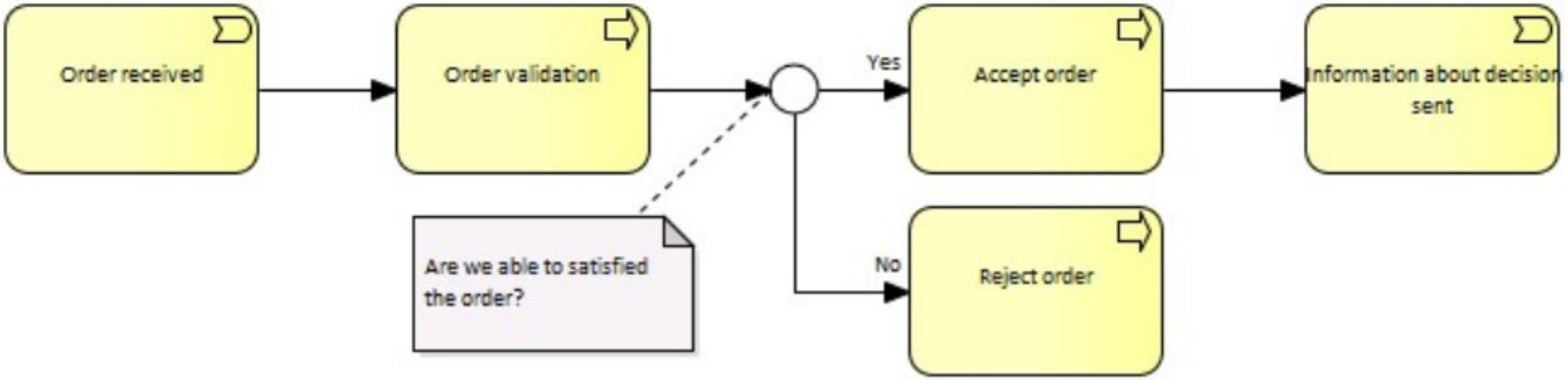


Element Udalosť v biznis vrstve (Event)

- **Typ**: **aktívny interný** (Behavior)
- **Definícia**:
  - Predstavuje zmenu stavu. Môže byť vyvolaná či zachytená v rámci organizácie či mimo nej.
- **Príklad použitia**:
  - Objednávka prijatá
  - Informácie o rozhodnutí odoslané
  - Dopis odoslaný

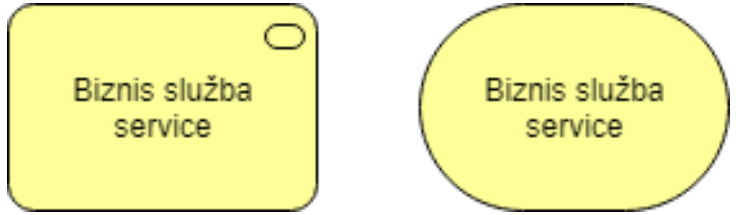


Element Udalosť v biznis vrstve (Event)

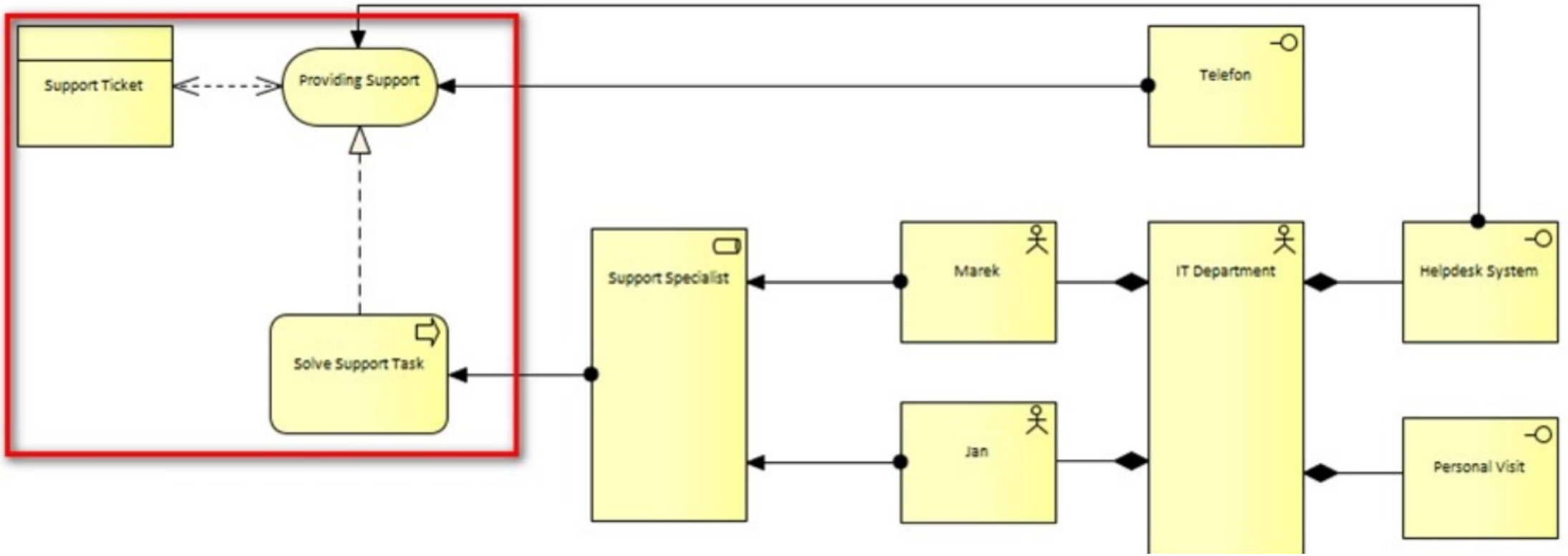


Element Služba v biznis vrstve (Service)

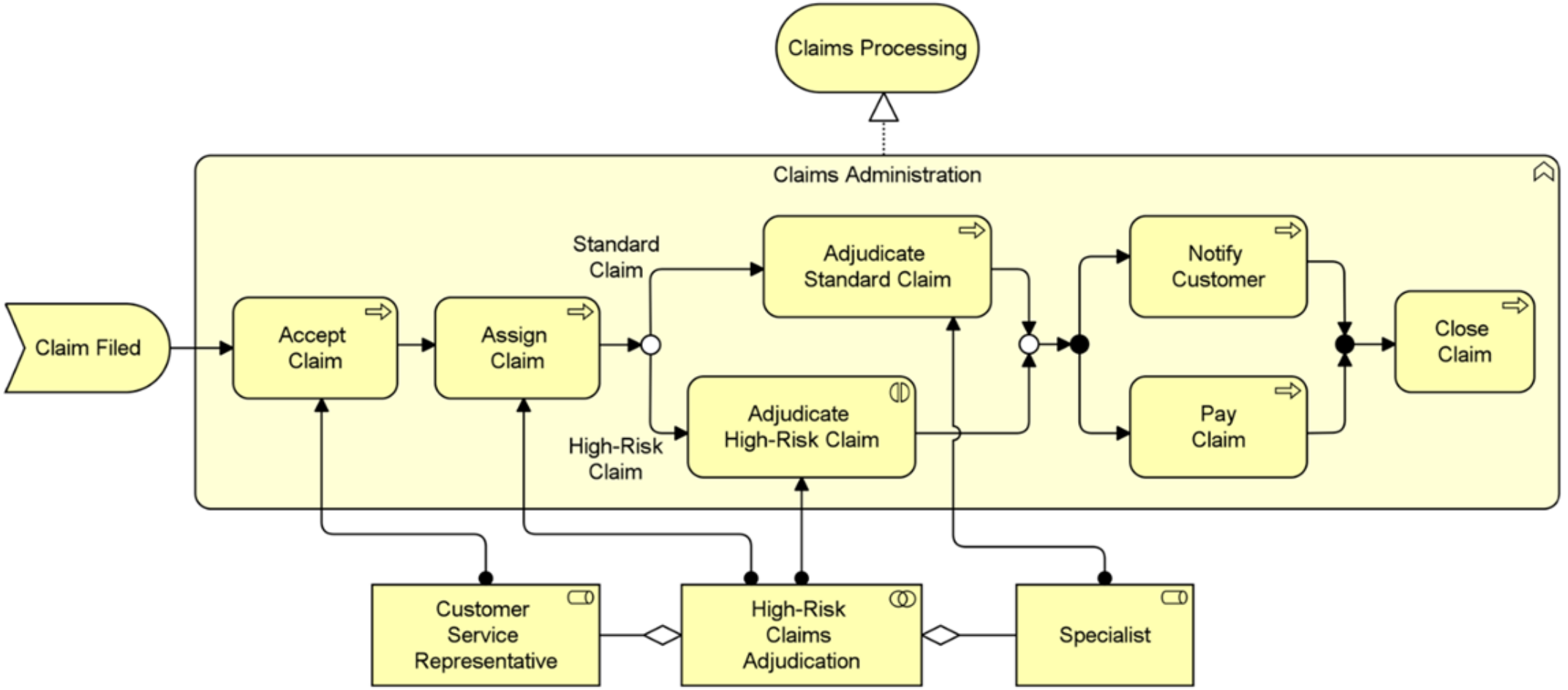
- **Typ**: **aktívny interný** (Behavior)
- **Definícia**:
  - Reprezentuje explicitne definovanú službu či chovanie vystavené do okolitého prostredia.
- **Príklad použitia**:
  - Poskytovanie podpory
  - Spracovanie pohľadávok



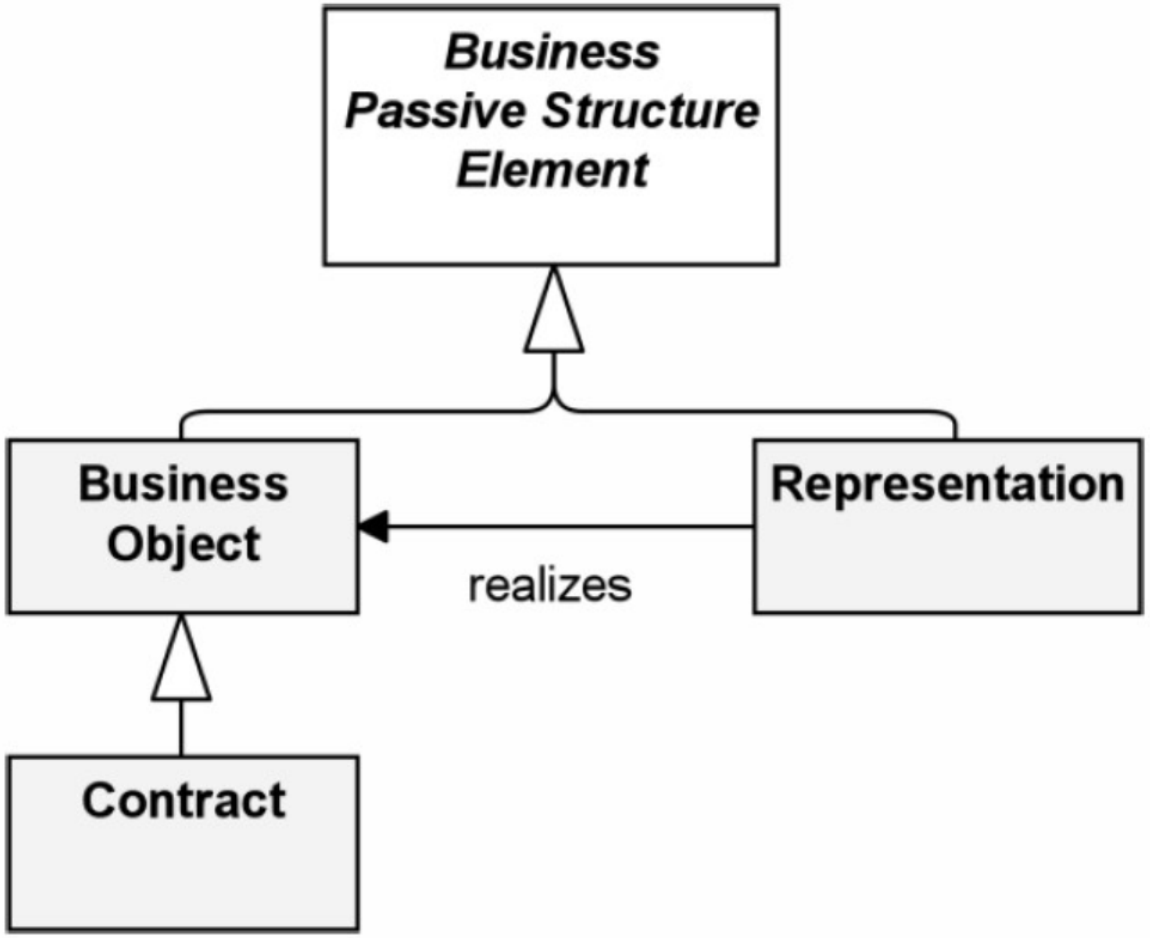
Element Služba v biznis vrstve (Service)



Element Služba v biznis vrstve (Service)

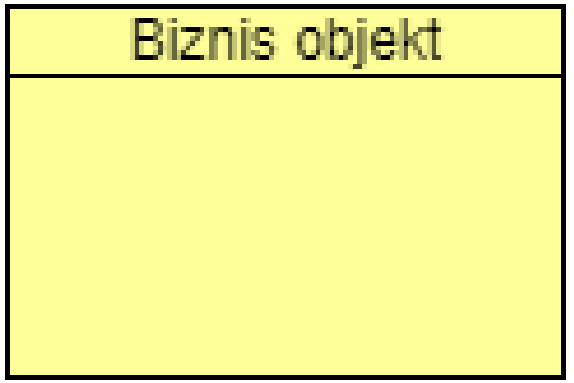


Element v biznis vrstve – Passive Structure

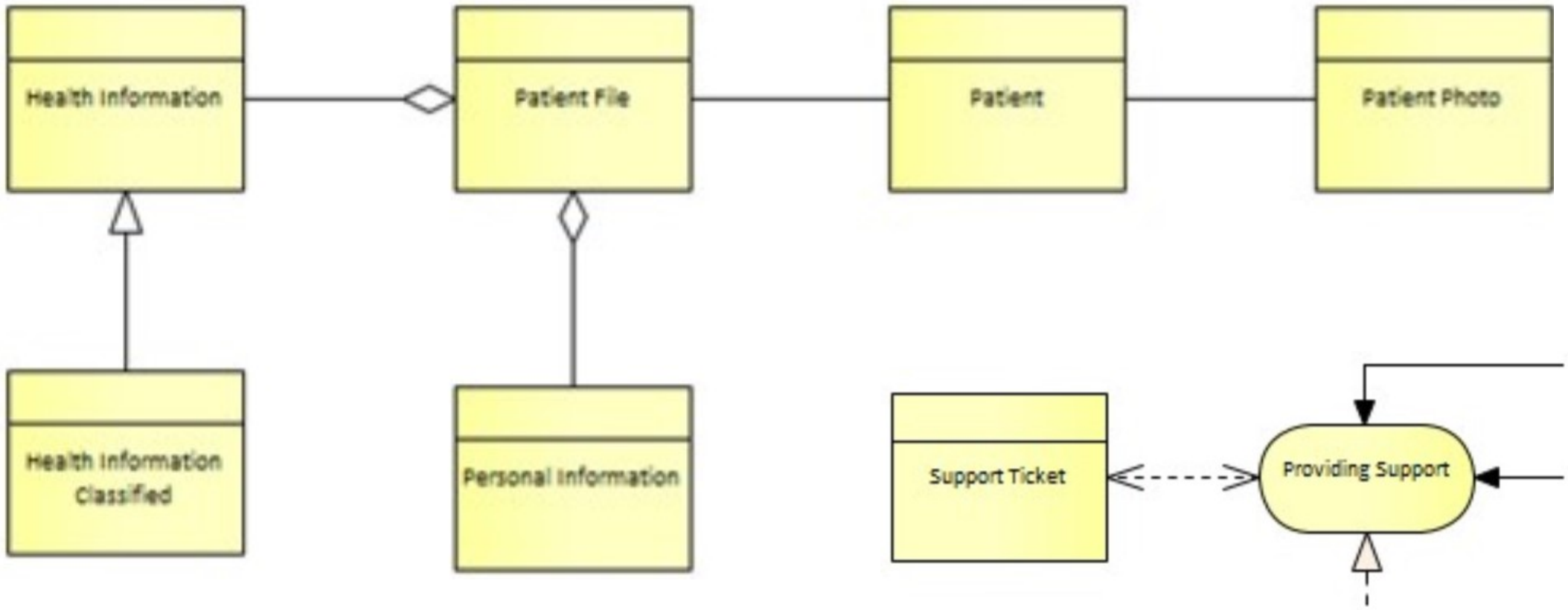


Element Objekt v biznis vrstve (Object)

- **Typ**: **pasívny** (Passive Structure)
- **Definícia**:
  - Reprezentuje koncept (Concept) použitý v rámci konkrétnej biznis domény.
- **Príklad použitia**:
  - faktúra
  - objednávka



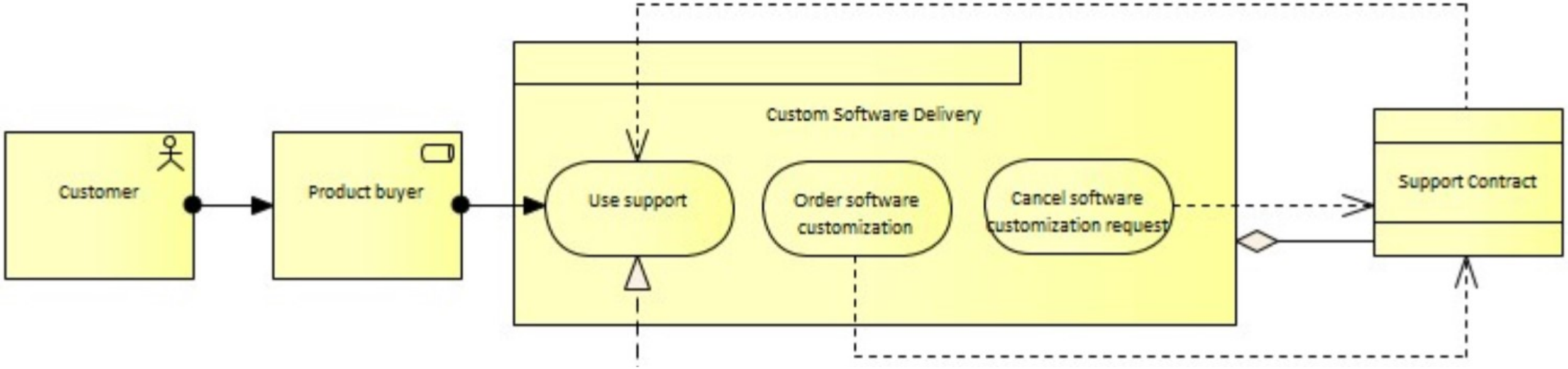
Element Objekt v biznis vrstve (Object)



Element Kontrakt v biznis vrstve (Contract)

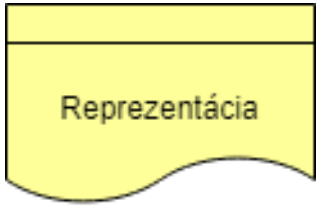
- **Typ**: **pasívny** (Passive Structure)
- **Definícia**:
  - Formálny či neformálny popis dohody medzi poskytovateľom a konzumentom (služieb), ktorá obsahuje práva a povinnosti súvisiace s produktom. Ďalej môžu stanoviť pravidlá pre ich komunikáciu či inú interakciu.
- **Príklad použitia**:
  - zmluvy o podpore
  - Spoločenská zmluva

Element Kontrakt v biznis vrstve (Contract)

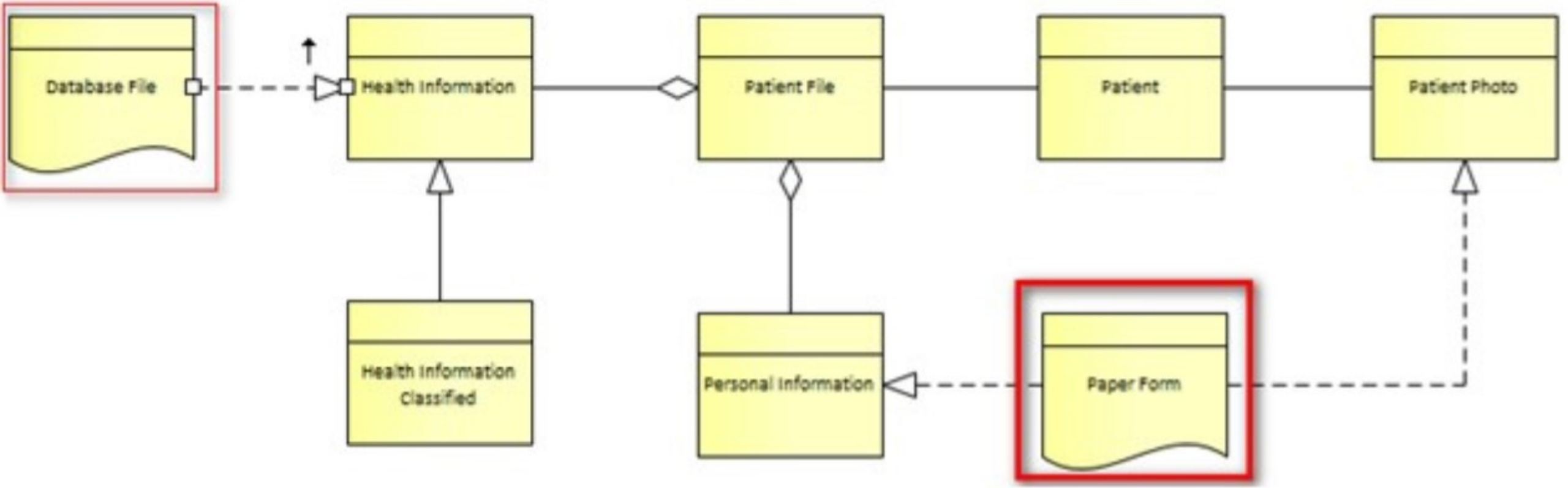


Element Reprezentácia v biznis vrstve (Representation)

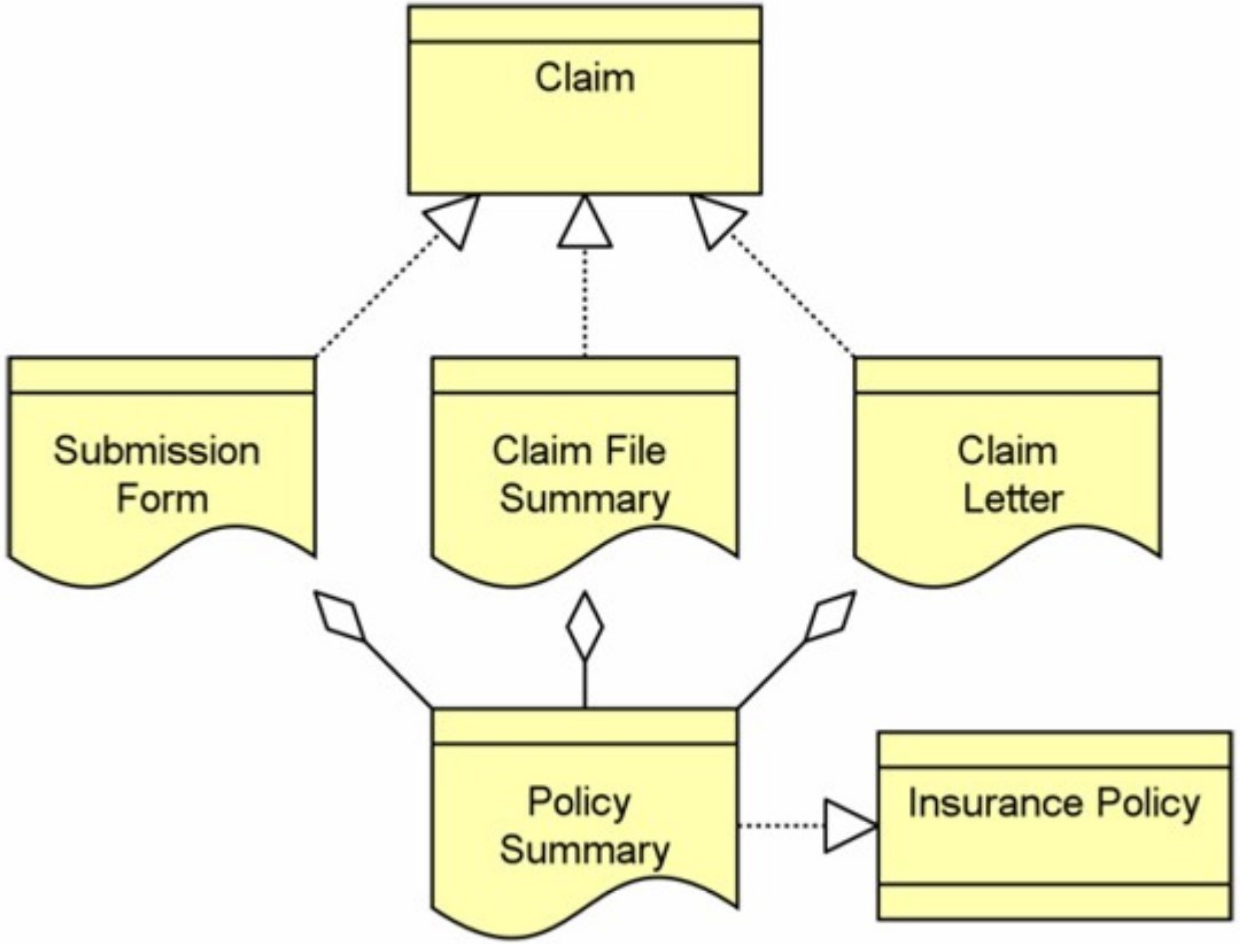
- **Typ**: **pasívny** (Passive Structure)
- **Definícia**:
  - Reprezentuje vnímateľnú (Perceptible) formu biznis objektov. Reprezentácia realizuje väzby pomocou väzby Realization.
- **Príklad použitia**:
  - Databázový súbor
  - Online formulár
  - Papierová forma



Element Reprezentácia v biznis vrstve (Representation)



Element Reprezentácia v biznis vrstve (Representation)



Čo je MVC Architektúra a na Čo je Dobrá?

Je MVC návrhový vzor alebo architektúra?

Systém

Môže mať viacero úrovní a rozsah:

1. **Funkcia** - implementácia funkcie
2. **Trieda** - implementácia triedy
3. **Projekt** - vzťahy medzi triedami
4. **Riešenie** - vzťahy medzi projektmi
5. **Systém** - vzťahy medzi riešeniami

System software vs. application software

System software	Application software
General-purpose software that manages basic system resources and processes	Software that performs specific tasks to meet user needs
Written in low-level assembly language or machine code	Written in higher-level languages, such as Python and JavaScript
Must meet specific hardware needs; interacts closely with hardware	Does not take hardware into account and doesn't interact directly with hardware
Installed at the same time as the OS, usually by the manufacturer	User or admin installs software when needed
Runs any time the computer is on	User triggers and stops the program
Works in the background and users don't usually access it	Runs in the foreground and users work directly with the software to perform specific tasks
Runs independently	Needs system software to run
Is necessary for the system to function	Isn't needed for the system to function

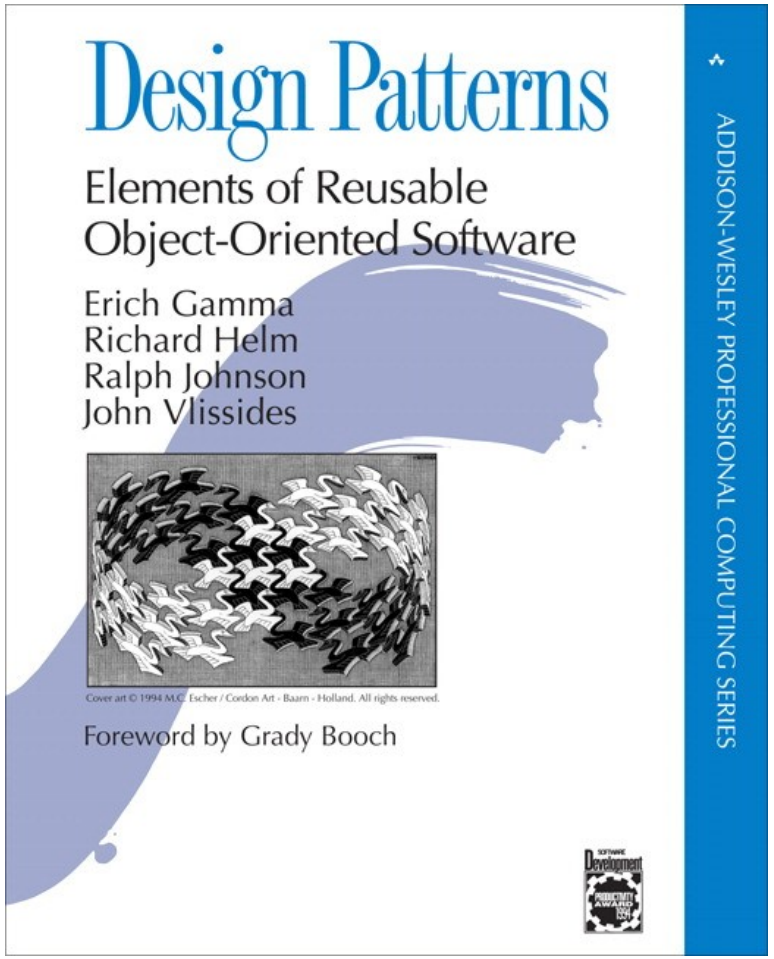
Návrhový Vzor (Design pattern)

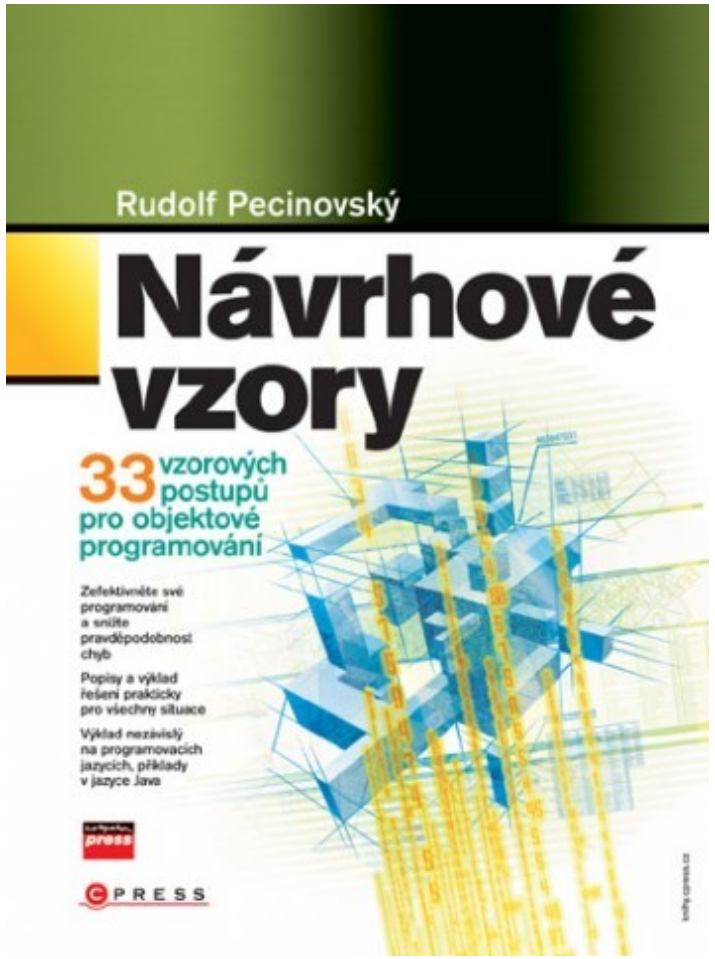
Všeobecne využiteľná predloha pre bežne sa vyskytujúci problém a kontexte dizajnu software

Predpis šablóny možného riešenia problému, ktorý je možné aplikovať vo viacerých situáciách

Vzory sú vypracované podľa Best Practises, aké môže programátor využiť pri riešení toho daného typu problému

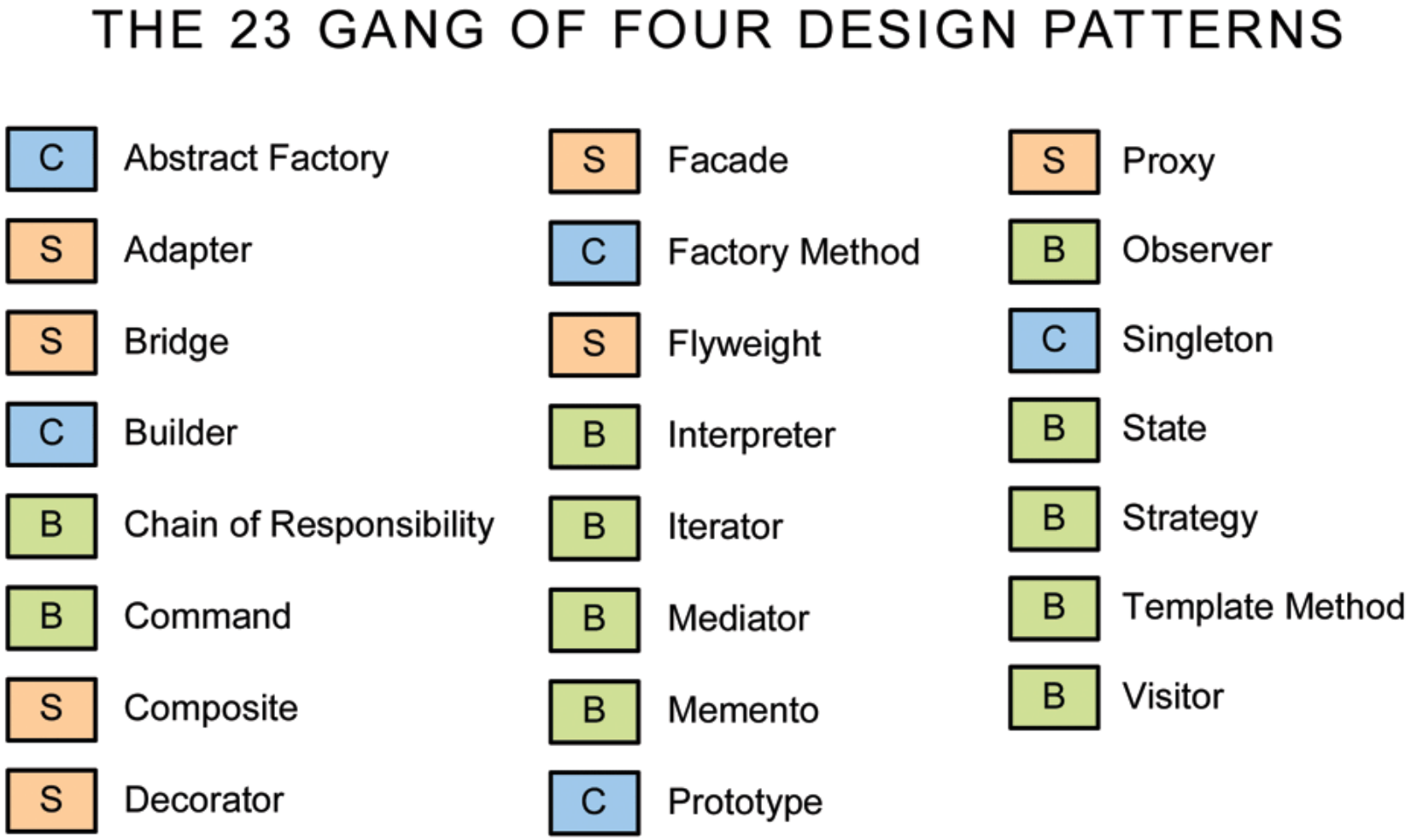
Gang of four - GOF





Click here to next software adventure

Koľko a aké sú návrhové vzory?



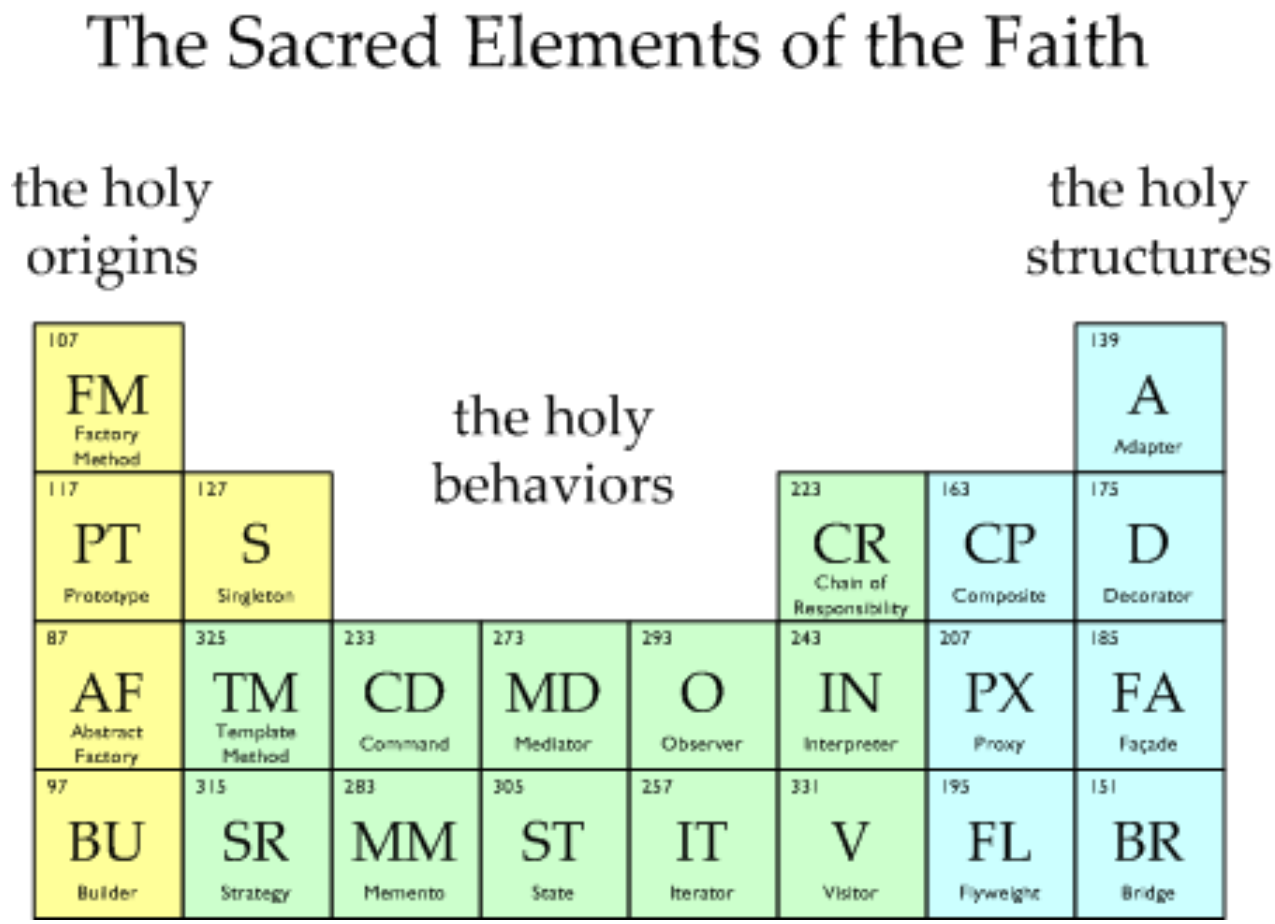
Podporné Vzory (Supporting Patterns)

Jednoduchá továrenská metóda - Simple Factory Method

Prepravka - Crate - Messenger

Prázdny objekt - Null Object

Knižničná trieda - Library class - Utility



C

Jednoduchá továrenská metóda
Simple Factory Method

Definuje statickú metódu nahradzujúcu konštruktor

Používa sa všade tam, kde potrebujeme získať odkaz na objekt

Pričom priame použitie konštruktora nie je moc ideálne

Rozdiel medzi Vzorom a Architektúrou

Návrhový vzor (Design pattern)

- Slúžia ako šablóny pre časť izolovaného problému
- Pomáha k lepšej štruktúre a udržiavateľnosti
- Zameriavajú na prvé **3 rozsahy**:

1. Funkcia - implementácia funkcie
2. Trieda - implementácia triedy
3. Projekt - vzťahy medzi triedami

Architektúra

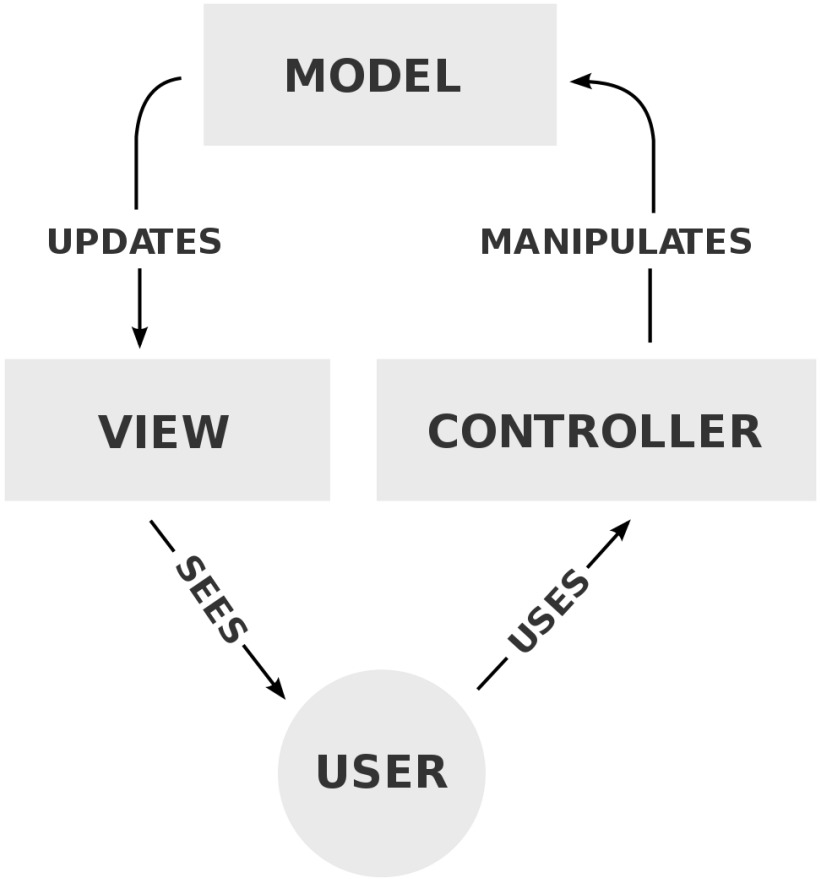
- Určená na spojenie všetkého v fungujúcom systéme
- Najvyššia úroveň návrhu riešenia ako celku
- Väčšinou premýšľate vo veľkom, keďže sa príliš nezachádzate do detailov rôznych komponentov
- Zaisťuje, že rôzne časti riešenia môžu spolu ľahko komunikovať buď priamo alebo cez sprostredkovateľa, týka sa to bezpečnosti, výkonu, nefunkčného nasadenia a systémových požiadaviek.
- Architektúra sa stará o **4 a 5 rozsahov** a niekedy aj 3

Model–View–Controller (MVC)

- Je **softvérový návrhový vzor**
- Bežne používaný na **vývoj používateľských rozhraní**, ktoré **rozdeľujú** súvisiacu **programovú logiku** do **3 vzájomne prepojených prvkov**
- Cieľom je **oddeliť interné reprezentácie informácií** od spôsobov, akými sú informácie prezentované používateľovi a akceptované od používateľa
- Tradične používaný pre **desktopové grafické používateľské rozhrania** (GUI) a rovnako aj pri **navrhovaní webových aplikácií**
- Väčšina populárnych programovacích jazykov má dostupné MVC frameworky, ktoré uľahčujú implementáciu vzoru (**Spring** a **Liferay MVC**)

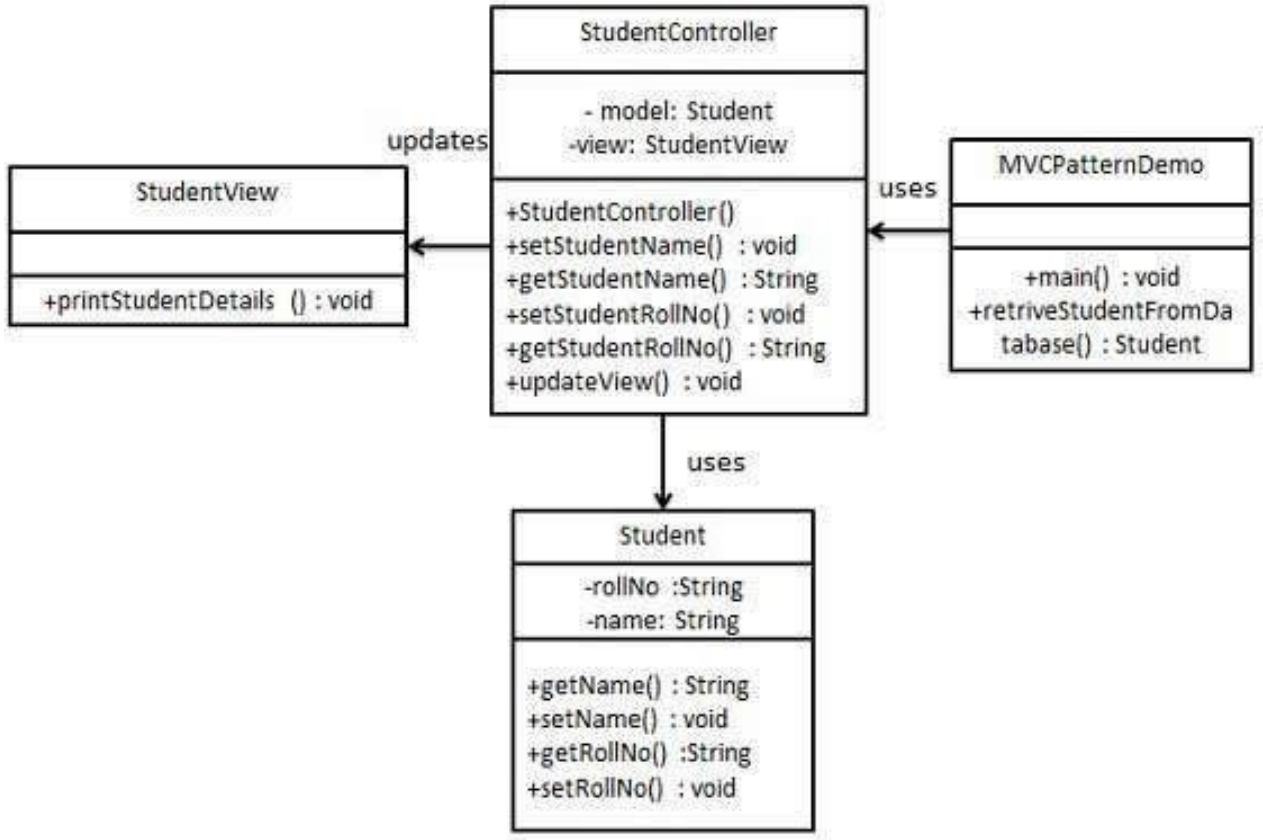
Model–view–controller (MVC)

1. **Model**: Hlavná/centrálna zložka vzoru. Je to dynamická dátová štruktúra aplikácie, **nezávislá od používateľského rozhrania**. Priamo **spravuje dáta, logiku a pravidlá aplikácie**.
2. **View**: **Akékoľvek reprezentácia informácií**, ako je graf, diagram alebo tabuľka. Sú možné viaceré zobrazenia rovnakých informácií, napríklad stĺpcový graf pre manažment a tabuľkové zobrazenie pre účtovníkov.
3. **Controller**: Prijíma **vstup** a **konvertuje** ho na **príkazy** pre **model** alebo **view**.

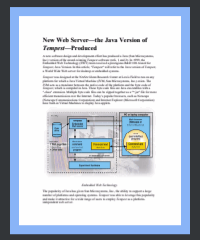


Model–view–controller (MVC)

- **Model** - predstavuje **objekt** alebo **JAVA POJO nesúci dáta**. Môže mať tiež **logiku** na aktualizáciu Controllera, ak sa zmenia jeho údaje.
- **View** – predstavuje **vizualizáciu údajov**, ktoré model obsahuje.
- **Controller** – pôsobí na model aj pohľad. **Riadi tok údajov** do objektu modelu a **aktualizuje zobrazenie** pri každej **zmene údajov**. Udržuje View a model oddelené.



POJO (Plain old Java object)


1

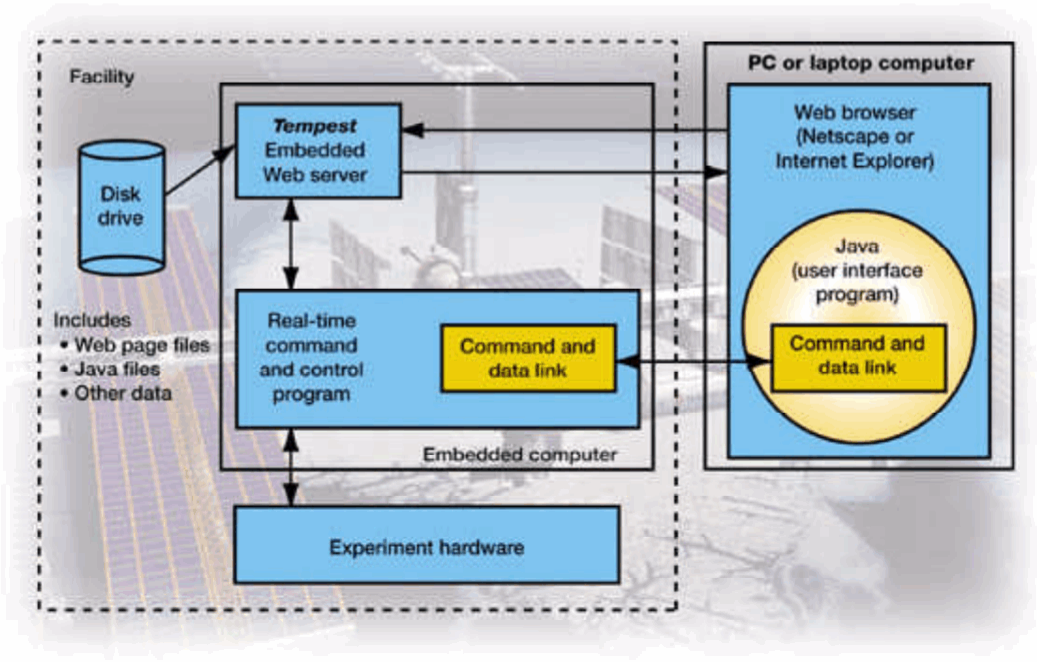

2


3

Tempest—Produced

A new software design and development effort has produced a Java (Sun Microsystems, Inc.) version of the award-winning *Tempest* software (refs. 1 and 2). In 1999, the Embedded Web Technology (EWT) team received a prestigious R&D 100 Award for *Tempest*, Java Version. In this article, "*Tempest*" will refer to the Java version of *Tempest*, a World Wide Web server for desktop or embedded systems.

Tempest was designed at the NASA Glenn Research Center at Lewis Field to run on any platform for which a Java Virtual Machine (JVM, Sun Microsystems, Inc.) exists. The JVM acts as a translator between the native code of the platform and the byte code of *Tempest*, which is compiled in Java. These byte code files are Java executables with a ".class" extension. Multiple byte code files can be zipped together as a "*.jar" file for more efficient transmission over the Internet. Today's popular browsers, such as Netscape (Netscape Communications Corporation) and Internet Explorer (Microsoft Corporation) have built-in Virtual Machines to display Java applets.



Embedded Web Technology.

JavaServer Faces Technology

Developed through the Java Community Process under JSR - 314, JavaServer Faces technology establishes the standard for building server-side user interfaces. With the contributions of the expert group, the JavaServer Faces APIs are being designed so that they can be leveraged by tools that will make web application development even easier. Several respected tools vendors were members of the JSR-314 expert group, which developed the JavaServer Faces 1.0 specification. These vendors are committed to supporting the JavaServer Faces technology in their tools, thus promoting the adoption of the JavaServer Faces technology standard.

Overview

JavaServer Faces technology includes:

- A set of APIs for representing UI components and managing their state, handling events and input validation, defining page navigation, and supporting internationalization and accessibility.
- A JavaServer Pages (JSP) custom tag library for expressing a JavaServer Faces interface within a JSP page.

Designed to be flexible, JavaServer Faces technology leverages existing, standard UI and web-tier concepts without limiting developers to a particular mark-up language, protocol, or client device. The UI component classes included with JavaServer Faces technology encapsulate the component functionality, not the client-specific presentation, thus enabling JavaServer Faces UI components to be rendered to various client devices. By combining the UI component functionality with custom renderers, which define rendering attributes for a specific UI component, developers can construct custom tags to a particular client device. As a convenience, JavaServer Faces technology provides a custom renderer and a JSP custom tag library for rendering to an HTML client, allowing developers of Java Platform, Enterprise Edition (Java EE) applications to use JavaServer Faces technology in their applications.

Ease-of-use being the primary goal, the JavaServer Faces architecture clearly defines a separation between application logic and presentation while making it easy to connect the presentation layer to the application code. This design enables each member of a web application development team to focus on his or her piece of the development process, and it also provides a simple programming model to link the pieces together. For example, web page developers with no programming expertise can use JavaServer Faces UI component tags to link to application code from within a web page without writing any scripts.

Developed through the Java Community Process under JSR - 314, JavaServer Faces technology establishes the standard for building server-side user interfaces. With the contributions of the expert group, the

Welcome to the Apache MyFaces Project

Apache MyFaces is a project of the Apache Software Foundation, and hosts several sub-projects relating to the JavaServer™ Faces (JSF) technology.

Projects

- MyFaces Core | Implementation of the JSF specification
- Apache Tobago | A component library

Inactive Projects (Maintenance Mode)

- MyFaces Commons | Utilities like components, converters, validators
- MyFaces Tomahawk | A component library
- MyFaces Trinidad | A component library (former Oracle ADF-Faces)
- MyFaces Orchestra | Utility library based on Spring
- MyFaces Extensions Validator | Validation framework based on annotations
- MyFaces Extensions CDI | Utility library based on CDI
- MyFaces Extensions Scripting | Adds scripting and rapid prototyping (hot deployment) to JSF
- MyFaces Portlet Bridge | Bridge between Portlets and JSF

Frameworks, JVM Internals, Tools

Java's evolution into 2022: The state of the four big initiatives

Nicolai Parlog | February 19, 2022
Developer Advocate, Oracle

This year will continue the evolution of projects Valhalla, Panama, Loom, and Amber in JDK 18 and JDK 19.

Download a PDF of this article

This will be a significant year for Java. As is well known, Java 18 (as JDK 18) will be generally available on March 22, 2022, and Java 19 will be generally available in September. Looking at the bigger picture of the specific JSRs, this year in Java will continue the evolution of four main initiatives: projects Valhalla, Panama, Loom, and Amber.

In this article, you will see the goals of each of these projects, their current state (as of January 2022), and their short-term plans.

Project Valhalla

Project Valhalla has two goals: an obvious one and a subtle one. The obvious goal is to introduce a new kind of type that "codes like a class, works like an int," as the mission statement says. The subtle goal is to heal the rift in Java's type system that separates reference types from primitive types; this will allow generification over *all* types.

Yes, you'll be able to create an `ArrayList<int>` that's backed by an `int[]`, with no boxing needed.

Valhalla launched in 2014 right after Java 8 was released and has spent much of that time in an exploratory phase during which the people behind the project, led by Brian Goetz, experimented with various proof-of-concept implementations.