Kapitola 5 – Modelovanie systému

Systémové modelovanie

- Systémové modelovanie je proces vývoja abstraktných modelov systému, pričom každý model predstavuje iný pohľad na alebo perspektívu daného systému.
- Systémové modelovanie znamená reprezentovať systém pomocou nejakého druhu grafického zápisu, ktorý je v súčasnosti takmer vždy založený na zápisoch Unified Modeling Language (UML).
- Modelovanie systému pomáha analytikovi pochopiť funkčnosť systému a modely sa používajú na komunikáciu so zákazníkmi.

Existujúce a plánované modely systémov

- Pri navrhovaní požiadaviek sa používajú modely existujúceho systému. Pomáhajú objasniť, čo robí existujúci systém, a možno ich použiť ako základ pre diskusiu o jeho silných a slabých stránkach. Tie potom vedú k požiadavkám na nový systém.
- Modely nového systému sa používajú pri navrhovaní požiadaviek, aby pomohli vysvetliť navrhované požiadavky ostatným účastníkom systému. Inžinieri používajú tieto modely na diskusiu o návrhoch dizajnu a na dokumentáciu systému na implementáciu.
- V modelom riadenom inžinierskom (MDE) procese je možné z modelu systému vygenerovať úplnú alebo čiastočnú implementáciu systému.

UML

- **Unified** (Booch , Rumbaugh , Jacobson)
  Modeling (vizuálny, grafický)
  Language (gramatika, syntax, sémantika)

- Definícia Object Management Group (OMG):
  - Unified Modeling Language je grafický jazyk na vizualizáciu, špecifikáciu, konštrukciu a dokumentáciu artefaktov softvérovo náročného systému. UML ponúka štandardný spôsob písania plánov systému vrátane koncepčných vecí, ako sú obchodné procesy a systémové funkcie, ako aj konkrétnych vecí, ako sú príkazy programovacieho jazyka, schémy databázy a opakovane použiteľné softvérové komponenty .

História UML

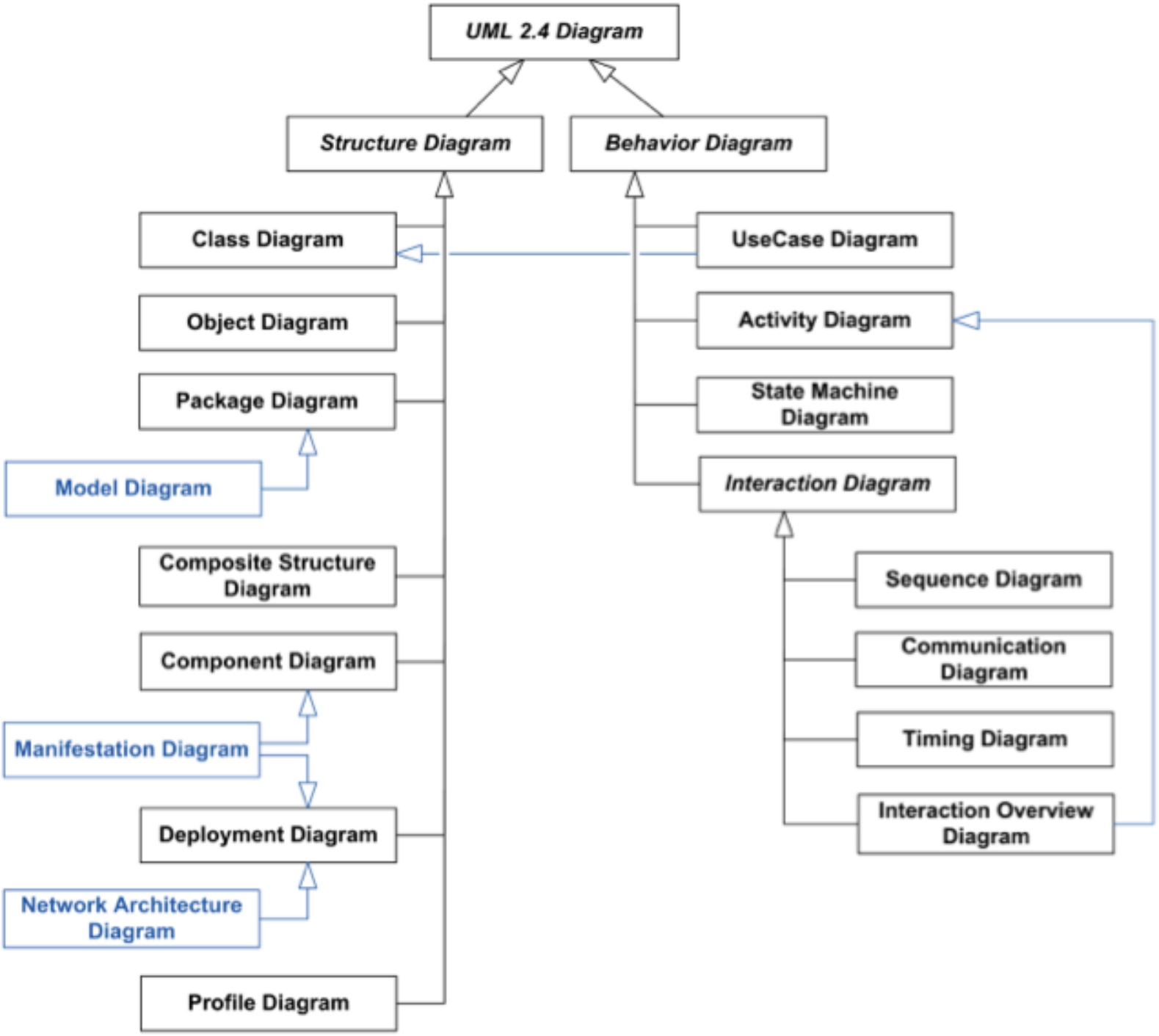
- 1996 : verzia 0.9 – pridali sa IBM, HP, MS, Oracle,...
- 1997: UML štandardizované spoločnosťou Object Management Group (OMG), verzie 1.0 a 1.1
- 1998-2001: verzie 1.2, 1.3, 1.4 – malé zmeny
- ISO/IEC 19501:2005 (UML verzia 1.4.2)
- 2005: verzia 2.0
- Aktuálna verzia 2.5.1 od decembra 2017

Systémové perspektívy

- Externá perspektíva, kde modelujete kontext alebo prostredie systému z vonkajšieho pohľadu.
- Interakčná perspektíva, kde modelujete interakcie medzi systémom a jeho prostredím alebo medzi komponentmi systému.
- Štrukturálna perspektíva, kde modelujete organizáciu systému alebo štruktúru údajov, ktoré systém spracováva.
- Perspektíva správania, kde modelujete dynamické správanie systému a ako reaguje na udalosti.

Typy diagramov UML

- Diagramy aktivít (činností), ktoré zobrazujú činnosti zahrnuté v procese alebo pri spracovaní údajov.
- Diagramy prípadov použitia, ktoré zobrazujú interakcie medzi systémom a jeho prostredím.
- Sekvenčné diagramy, ktoré zobrazujú interakcie medzi aktérmi a systémom a medzi komponentmi systému.
- Diagramy tried, ktoré zobrazujú triedy objektov v systéme a asociácie medzi týmito triedami.
- Stavové diagramy, ktoré ukazujú, ako systém reaguje na vnútorné a vonkajšie udalosti.


UML 2.4 Diagram
Structure Diagram
Behavior Diagram
Class Diagram
Object Diagram
Package Diagram
Model Diagram
Composite Structure Diagram
Component Diagram
Manifestation Diagram
Deployment Diagram
Network Architecture Diagram
Profile Diagram
UseCase Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Sequence Diagram
Communication Diagram
Timing Diagram
Interaction Overview Diagram

Použitie grafických modelov

- Ako prostriedok na uľahčenie diskusie o existujúcom alebo navrhovanom systéme
  - Neúplné a čiastočne nesprávne modely sú v poriadku, pretože ich úlohou je podporovať diskusiu.
- Ako spôsob dokumentácie existujúceho systému
  - Modely by mali byť presnou reprezentáciou systému, ale nemusia byť úplné.
- Ako podrobný popis systému, ktorý možno použiť na vygenerovanie implementácie systému
  - Modely musia byť správne aj úplné

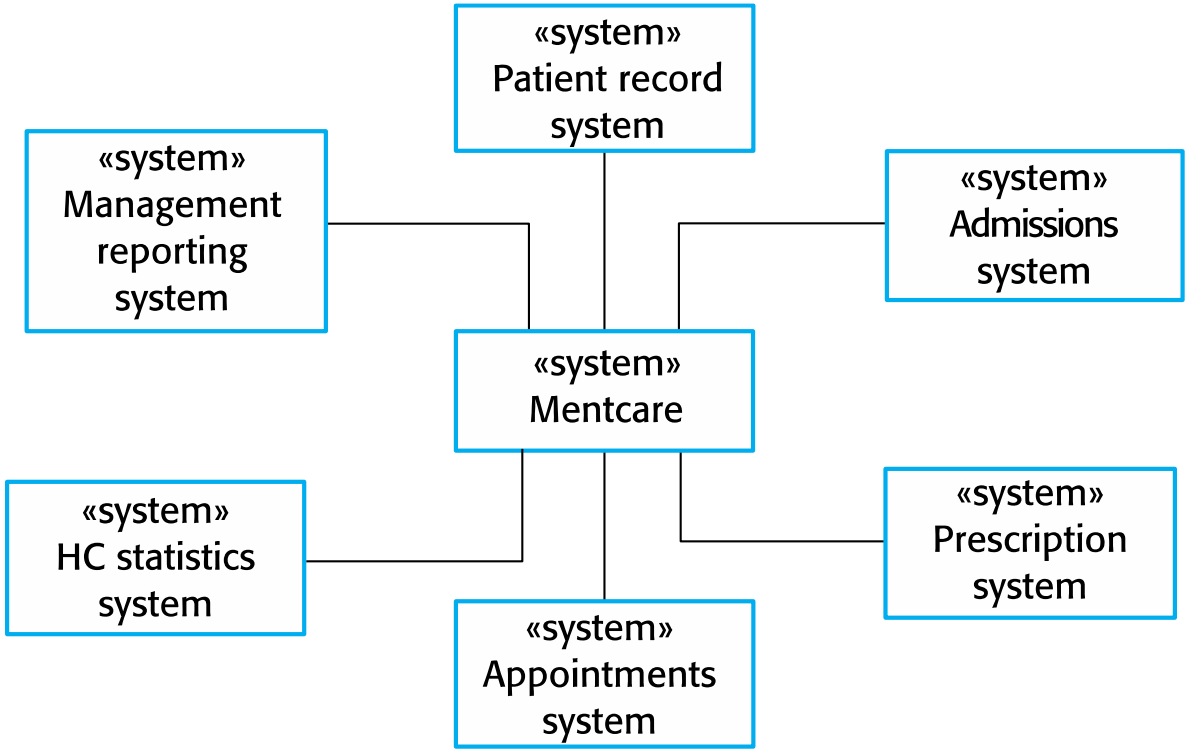
(1) Externá perspektíva - Kontextové modely

- Kontextové modely sa používajú na ilustráciu operačného kontextu systému – ukazujú, čo leží mimo hraníc systému.
- Sociálne a organizačné obavy môžu ovplyvniť rozhodnutie o umiestnení hraníc systému.
- Architektonické modely ukazujú systém a jeho vzťah s inými systémami.

Hranice systému

- Hranice systému sú stanovené tak, aby definovali, čo je vnútri a čo je mimo systém.
  - Zobrazujú ďalšie systémy, ktoré sa používajú alebo závisia od vyvíjaného systému.
- Poloha systémovej hranice má zásadný vplyv na systémové požiadavky.
- Definovanie systémovej hranice je politický úsudok
  - Môžu existovať tlaky na vytvorenie systémových hraníc, ktoré zvyšujú/znižujú vplyv alebo pracovné zaťaženie rôznych častí organizácie.

Kontext systému Mentcare

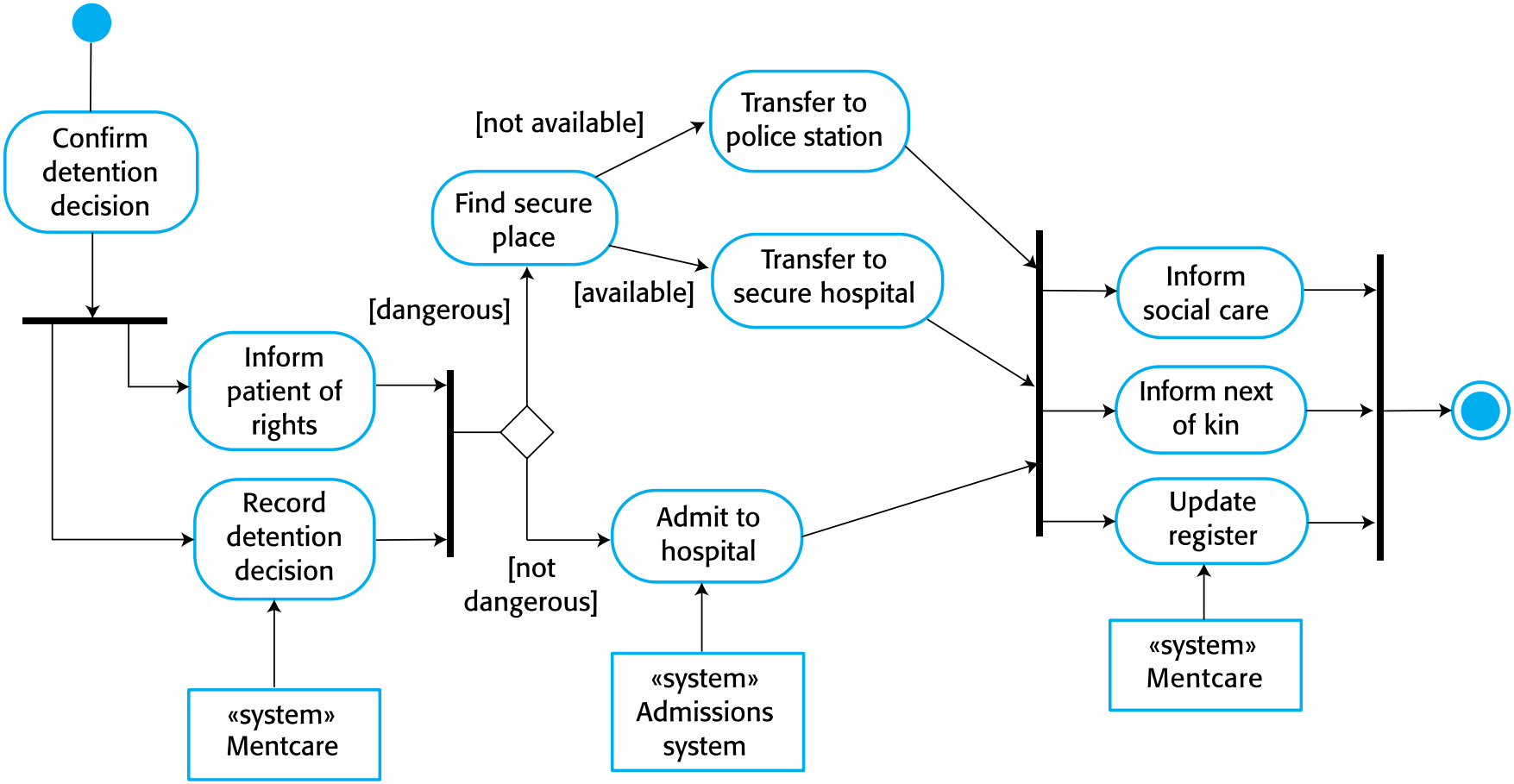


- Box-and-line diagram (škatuľky a čiary)
- UML diagram tried – kde triedy sú napr. <<system>>

(2) Procesná perspektíva

- Kontextové modely jednoducho ukazujú ostatné systémy v prostredí, nie to, ako sa vyvíjaný systém používa v tomto prostredí.
- Procesné modely odhaľujú, ako sa vyvíjaný systém používa v širších obchodných procesoch.
- Na definovanie sa môžu použiť diagramy aktivít UML
  - Modely podnikových procesov (workflow).
  - Modely transformácií údajov (dataflow).

Procesný model nedobrovoľného zadržania



(3) Interakčné modely

- Modelovanie interakcie používateľov je dôležité, pretože pomáha identifikovať požiadavky používateľov.
- Modelovanie interakcie medzi systémami poukazuje na komunikačné problémy, ktoré môžu vzniknúť.
- Modelovanie interakcie komponentov nám pomáha pochopiť, či navrhovaná štruktúra systému pravdepodobne poskytne požadovaný výkon a spoľahlivosť systému.
- Na modelovanie interakcie možno použiť UML diagramy prípadov použitia a diagramy sekvencií a komunikácie .

Modelovanie prípadov použitia

- Prípady použitia boli pôvodne vyvinuté na podporu vyvolávania požiadaviek a teraz sú začlenené do UML.
- Každý prípad použitia predstavuje samostatnú úlohu, ktorá zahŕňa externú interakciu so systémom.
- Aktérmi v prípade použitia môžu byť ľudia alebo iné systémy.
- Znázorňované diagramom s cieľom poskytnúť prehľad prípadu použitia a v podrobnejšej textovej forme.

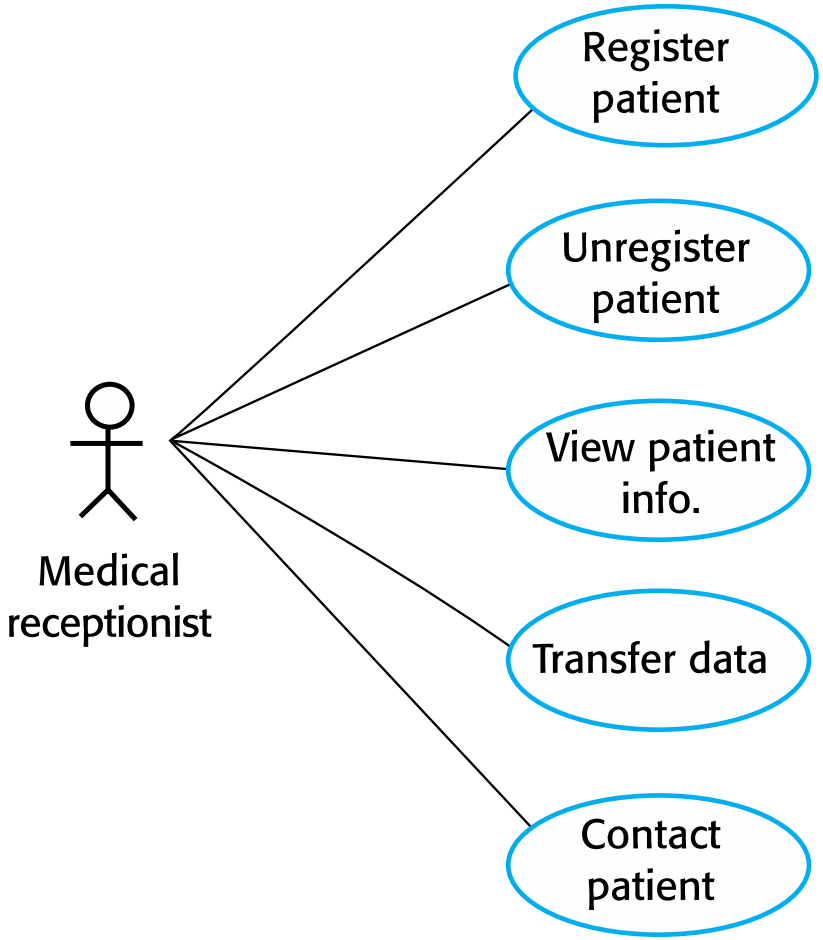
Prípad použitia "prenosu údajov"

✧ Prípad použitia v Mentcare systéme

Tabuľkový popis prípadu použitia "prenos údajov".

MHC -PMS: Prenos dát	
Herci	Lekárska recepčná, systém záznamov o pacientoch (PRS)
Popis	Recepčný môže preniesť údaje zo systému Mentcase do všeobecnej databázy záznamov o pacientoch, ktorú spravuje zdravotnícky úrad. Prenesené informácie môžu byť buď aktualizované osobné údaje (adresa, telefónne číslo atď.) alebo súhrn pacientovej diagnózy a liečby.
Údaje	Osobné údaje pacienta, súhrn liečby
Stimulácia	Užívateľský príkaz vydaný lekárskou recepciou
odpoveď	Potvrdenie, že PRS bola aktualizovaná
Komentáre	Recepčný musí mať príslušné bezpečnostné povolenia na prístup k informáciám o pacientovi a PRS .

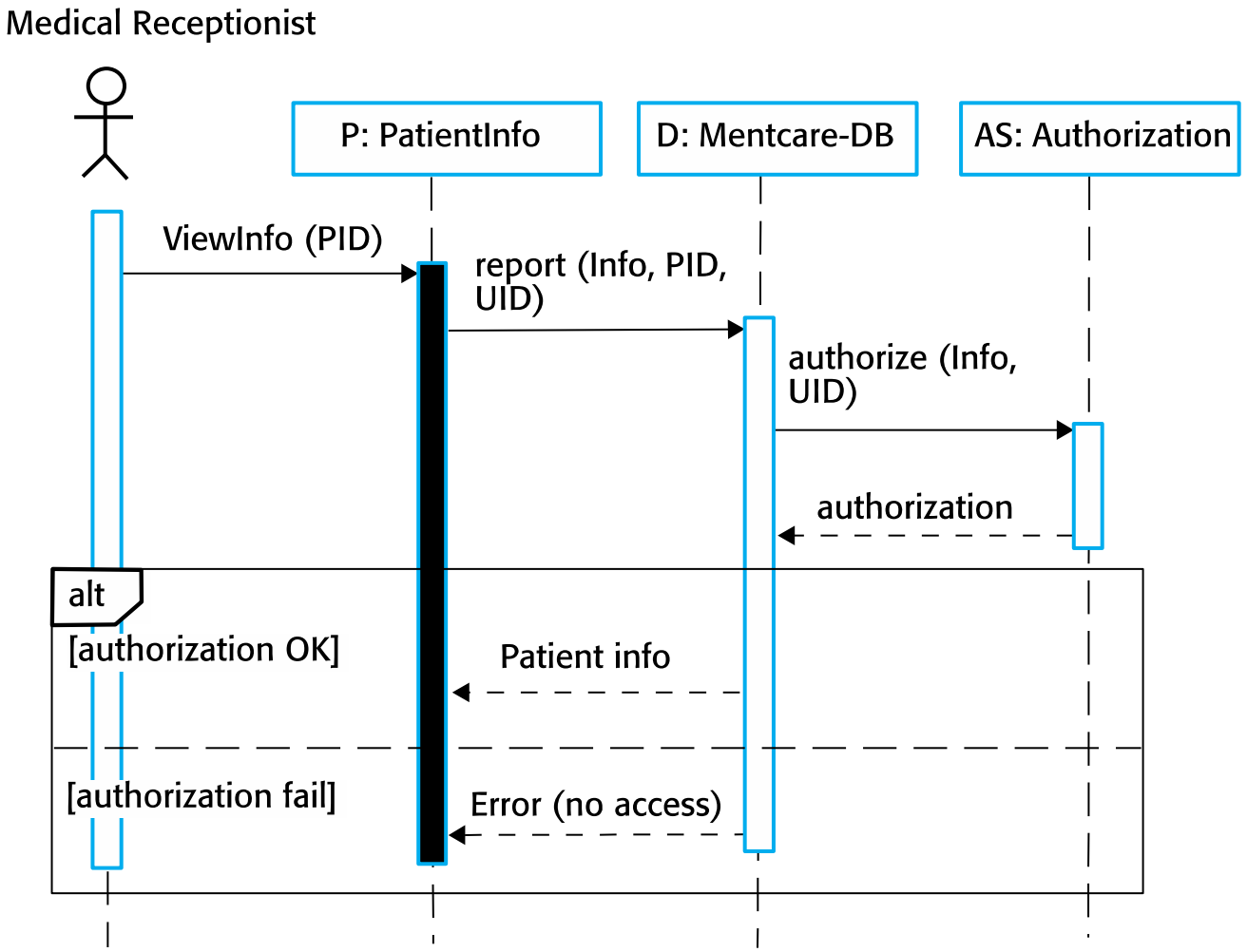
Prípady použitia v systéme Mentcare zahŕňajúce hráča „recepčný“

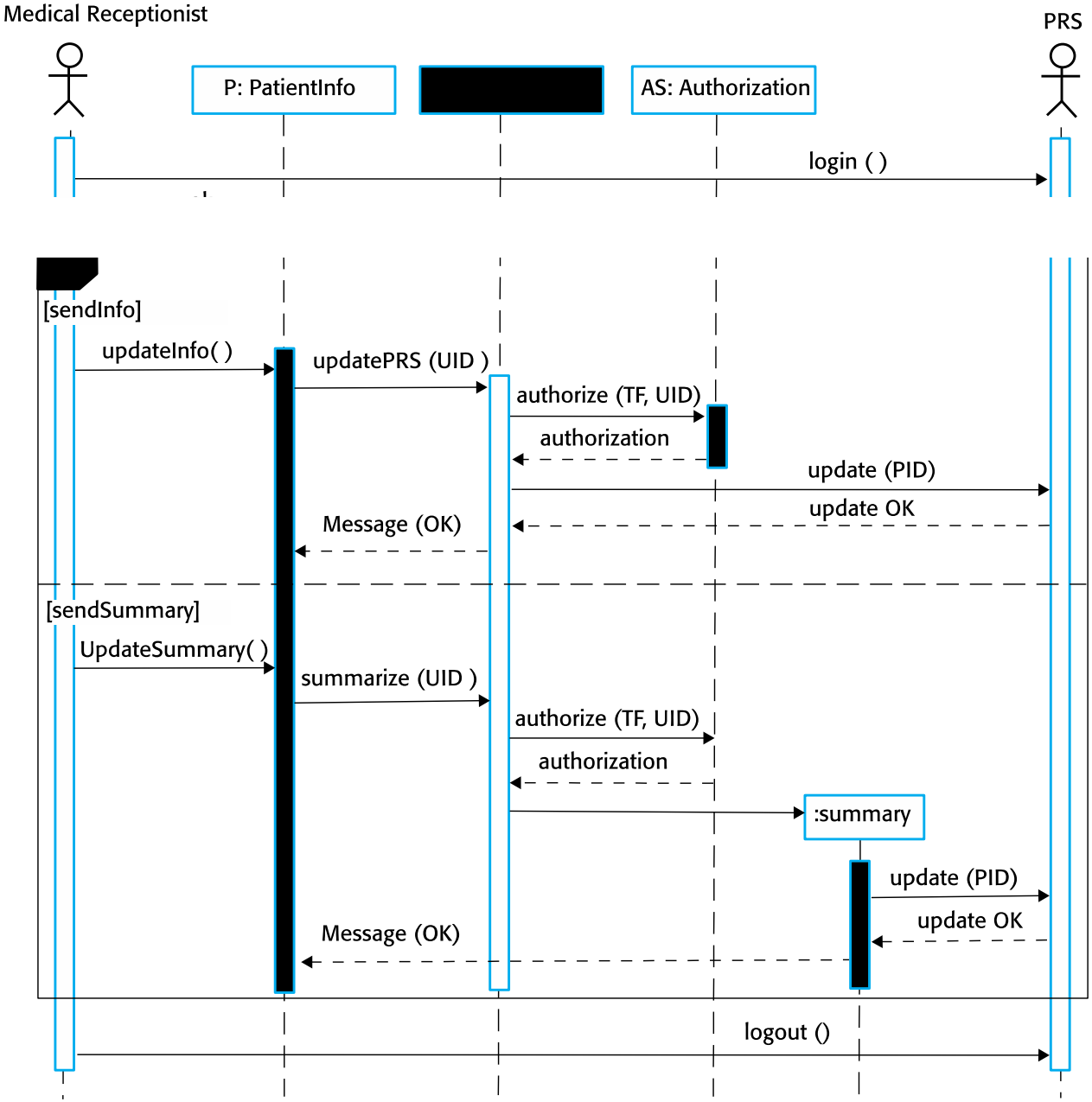


Sekvenčné diagramy

- Sekvenčné diagramy sú súčasťou UML a používajú sa na modelovanie interakcií medzi aktérmi a objektmi v rámci systému.
- Sekvenčný diagram zobrazuje postupnosť interakcií, ktoré prebiehajú počas konkrétneho prípadu použitia alebo inštancie prípadu použitia.
- Predmety a aktéri, ktorých sa to týka, sú uvedené v hornej časti diagramu, pričom z nich je vertikálne nakreslená bodkovaná čiara.
- Interakcie medzi objektmi sú označené šípkami s poznámkami.

Sekvenčný diagram pre zobrazenie informácií o pacientovi





Sekvenčný diagram pre prenos dát

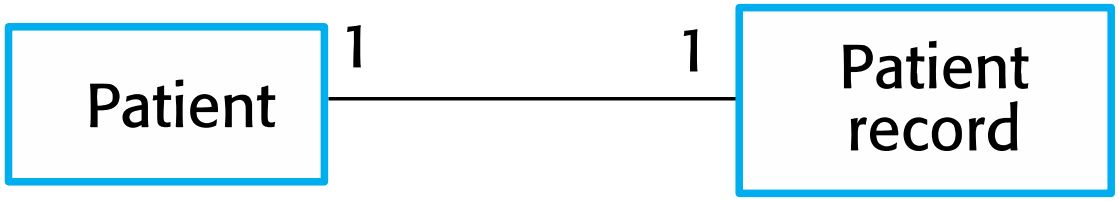
(4) Štrukturálne modely

- Štrukturálne modely softvéru zobrazujú organizáciu systému z hľadiska komponentov, ktoré tvoria tento systém, a ich vzťahov.
- Štrukturálne modely môžu byť statické modely, ktoré zobrazujú štruktúru návrhu systému, alebo dynamické modely, ktoré zobrazujú organizáciu systému pri jeho vykonávaní.
- Štrukturálne modely systému vytvárate, keď diskutujete a navrhujete architektúru systému.

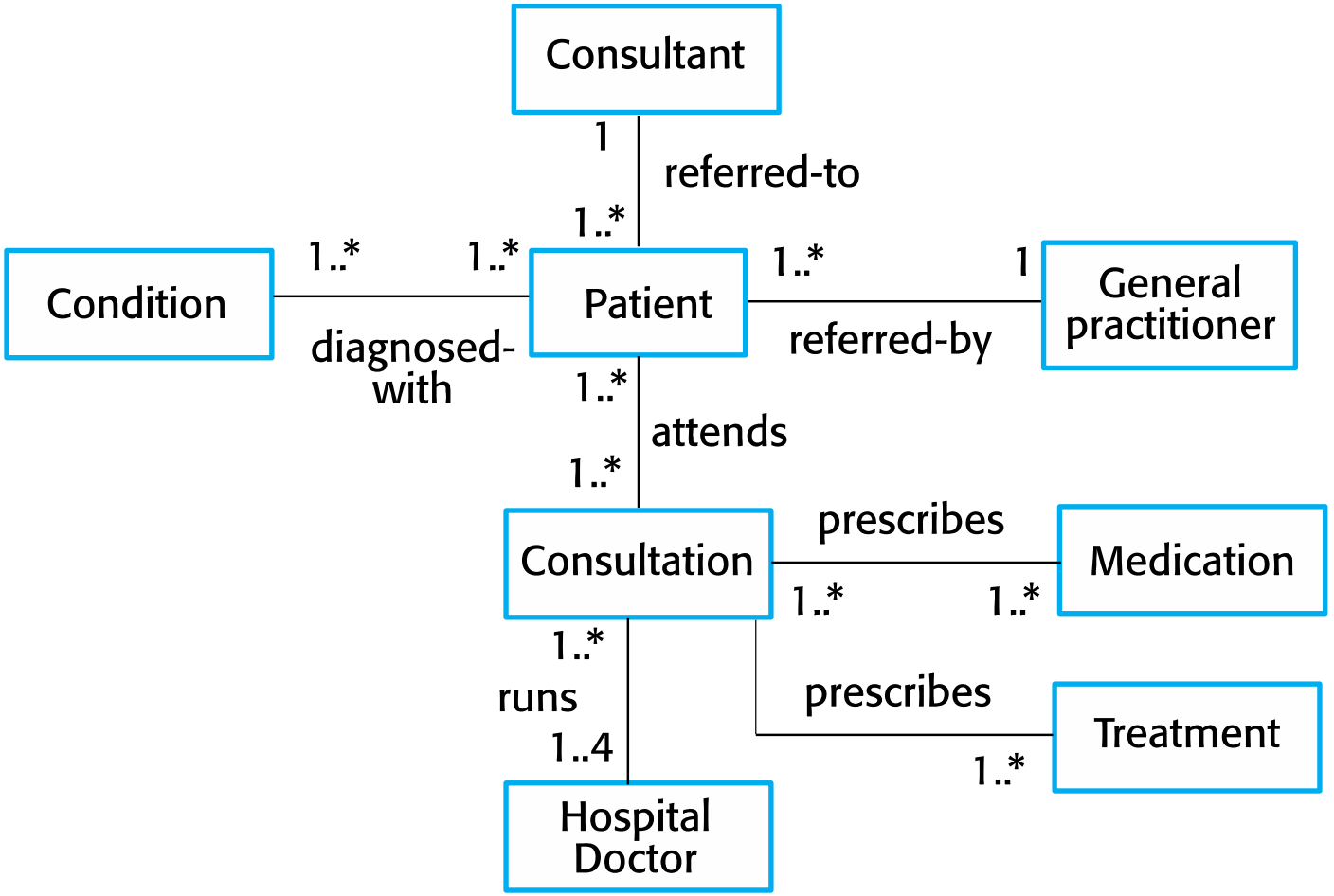
Diagramy tried

- Diagramy tried sa používajú pri vývoji objektovo orientovaného modelu systému na zobrazenie tried v systéme a asociácií medzi týmito triedami.
- Triedu objektov si možno predstaviť ako všeobecnú definíciu jedného druhu systémového objektu.
- Asociácia je prepojenie medzi triedami, ktoré naznačuje, že medzi týmito triedami existuje nejaký vzťah.
- Keď vyvíjate modely v počiatočných štádiách procesu softvérového inžinierstva, objekty predstavujú niečo v reálnom svete, ako je pacient, predpis, lekár atď.

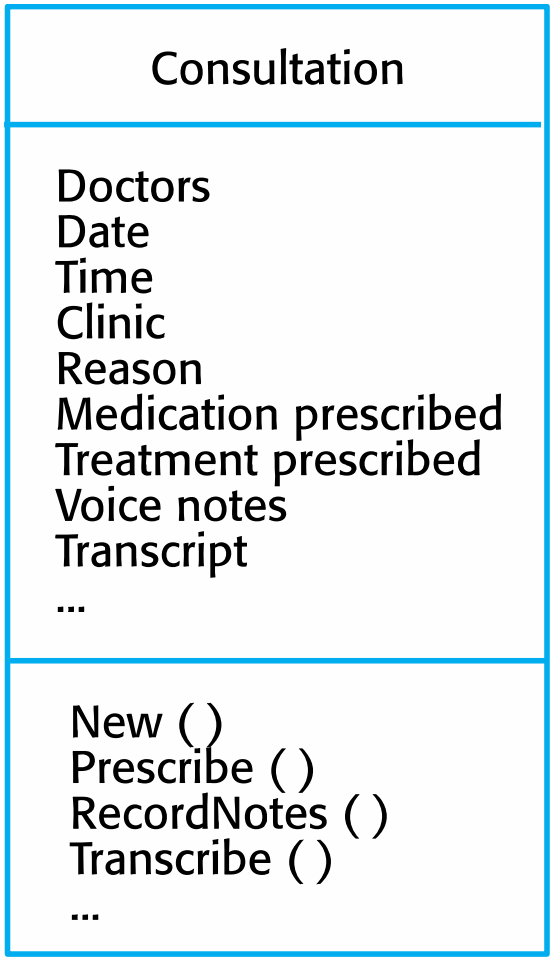
UML triedy a asociácie



Triedy a združenia v Mentcare



Konzultačná trieda



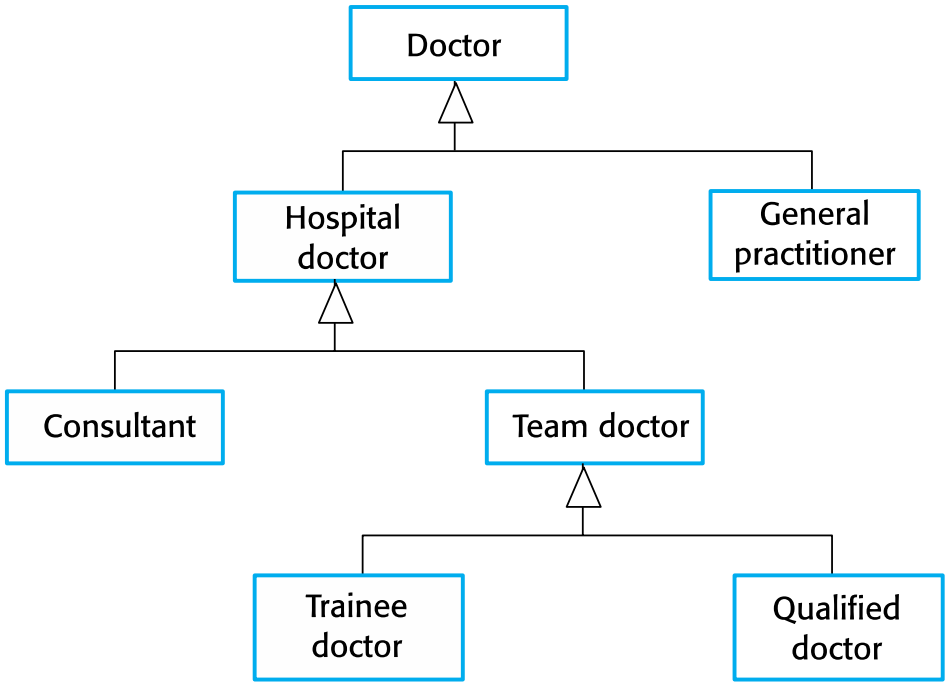
Zovšeobecnenie

- Zovšeobecňenie je každodenná technika, ktorú používame na zvládnutie zložitosti.
- Namiesto toho, aby sme sa učili podrobné charakteristiky každej entity, ktorú zažívame, zaraďujeme tieto entity do všeobecnejších tried (zvieratá, autá, domy atď.) a učíme sa charakteristiky týchto tried.
- To nám umožňuje odvodiť, že rôzni členovia týchto tried majú niektoré spoločné vlastnosti, napr. veveričky a potkany sú hlodavce.

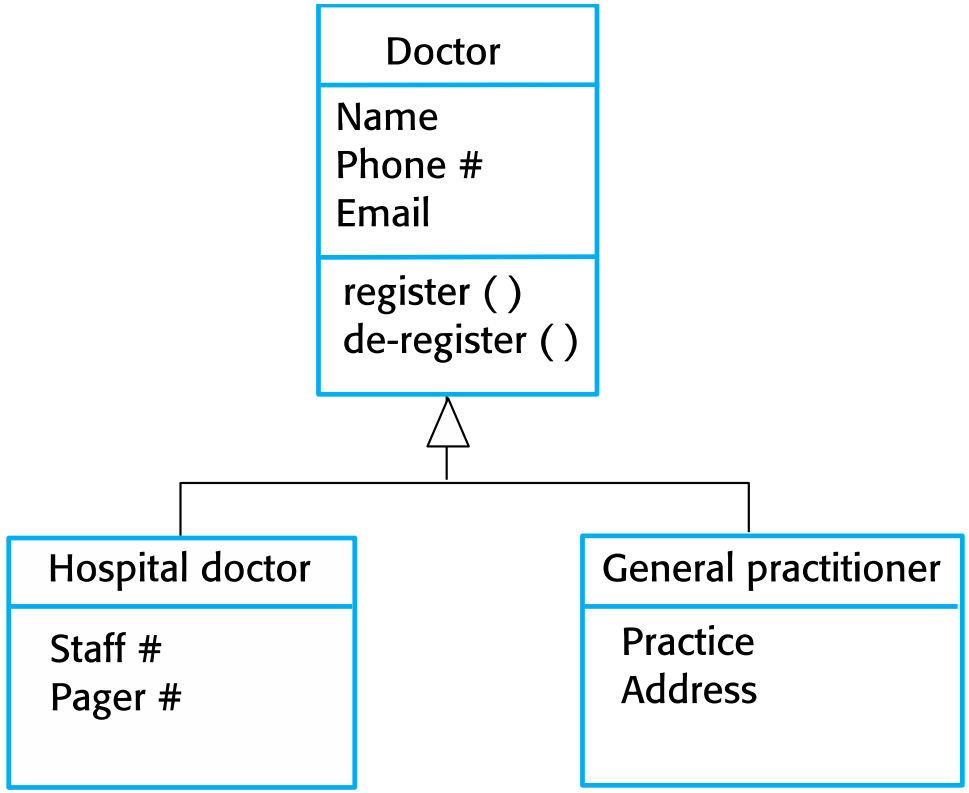
Zovšeobecnenie

- Pri modelovaní systémov je často užitočné preskúmať triedy v systéme, aby ste zistili, či existuje priestor na zovšeobecnenie. Ak sú navrhnuté zmeny, nemusíte sa pozerať na všetky triedy v systéme, aby ste zistili, či sa ich zmena týka.
- V objektovo orientovaných jazykoch, ako je Java, sa zovšeobecnenie implementuje pomocou mechanizmov dedičnosti tried zabudovaných do jazyka.
- Vo všeobecnosti sú atribúty a operácie spojené s triedami vyššej úrovne tiež spojené s triedami nižšej úrovne.
- Triedy nižšej úrovne sú podtriedy, ktoré zdedia atribúty a operácie zo svojich nadtried . Tieto triedy nižšej úrovne potom pridávajú špecifickejšie atribúty a operácie.

Hierarchia zovšeobecnenia (opak dedenia)



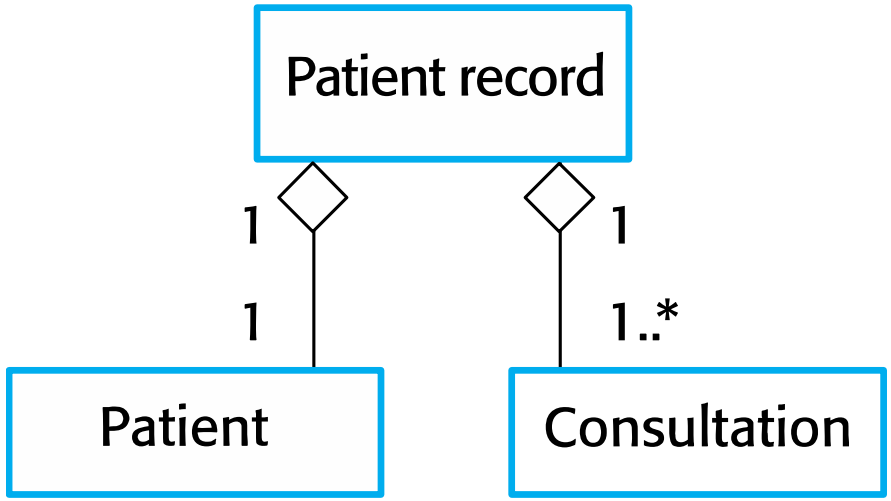
Hierarchia zovšeobecnenia s pridanými detailmi



Modely agregácie tried

- Agregačný model ukazuje, ako sa triedy, ktoré sú kolekciami, skladajú z iných tried.
- Agregačné modely sú podobné ako časť vzťahu v modeloch sémantických údajov.

Agregačná asociácia



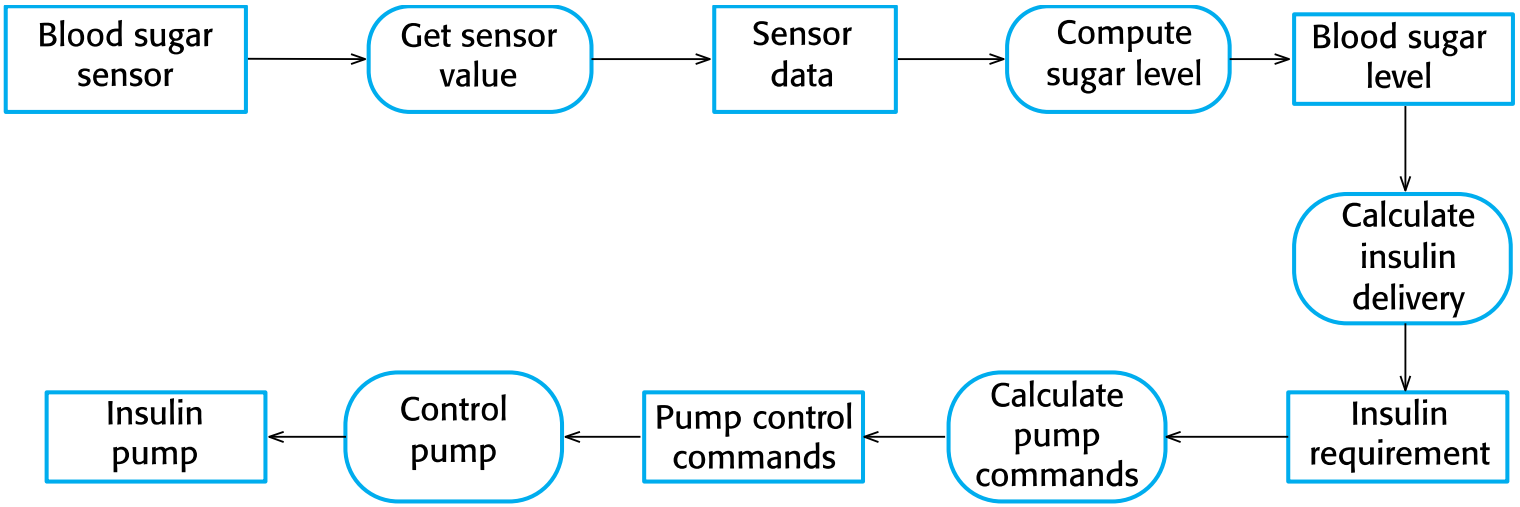
(5) Modely správania

- Behaviorálne modely (chovania sa, správania) sú modely dynamického správania sa systému pri jeho vykonávaní. Ukazujú, čo sa stane alebo čo sa má stať, keď systém zareaguje na podnet zo svojho okolia.
- Tieto stimuly si môžete predstaviť ako dva typy:
  - Údaje: Prichádzajú nejaké údaje, ktoré musí systém spracovať.
  - Udalosti: Nastane nejaká udalosť, ktorá spustí spracovanie systémom. Udalosti môžu mať súvisiace údaje, aj keď to nie je vždy tak.

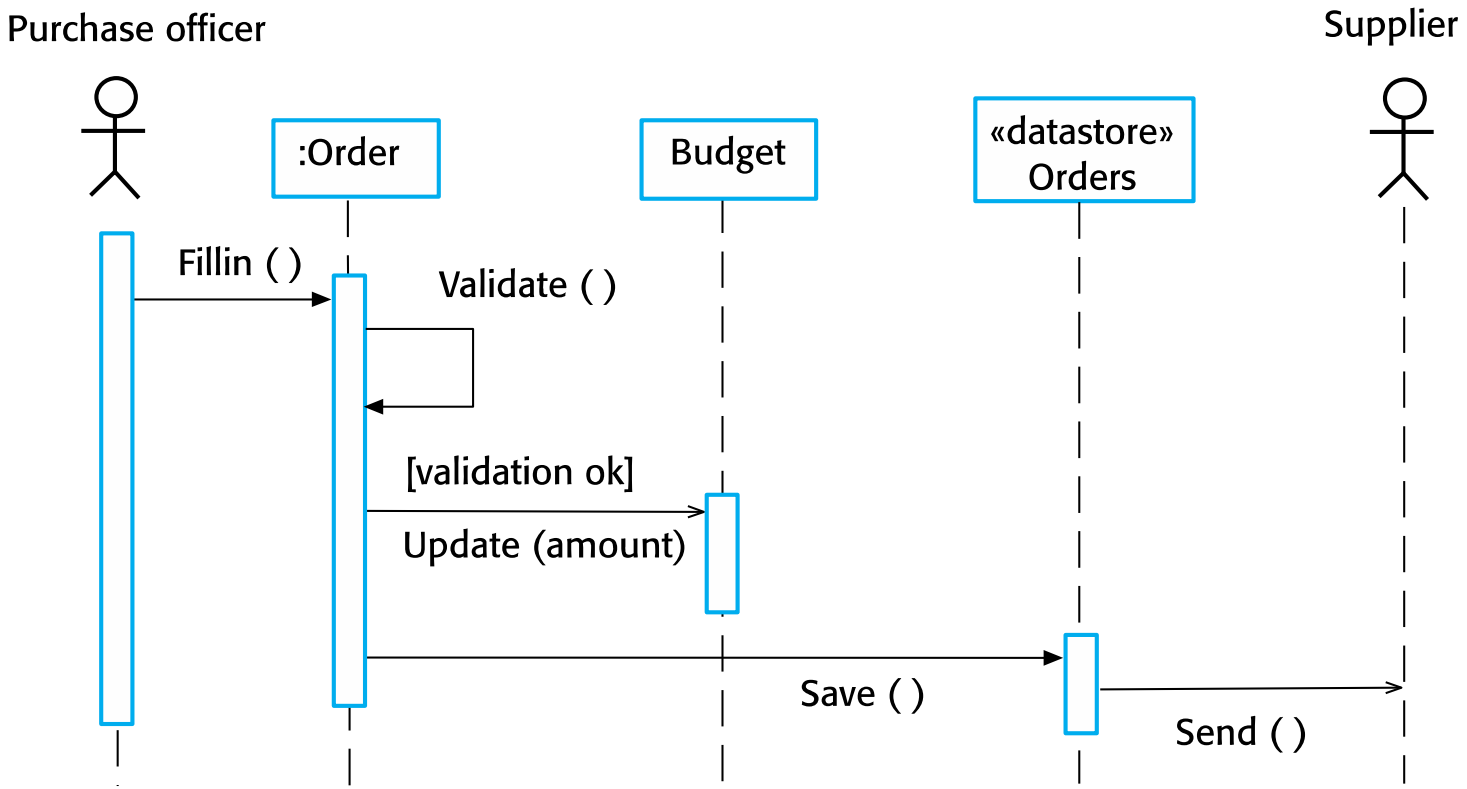
Modelovanie založené na údajoch

- Mnohé obchodné systémy sú systémy na spracovanie údajov, ktoré sú primárne poháňané údajmi. Sú riadené vstupom údajov do systému s relatívne malým externým spracovaním udalostí.
- Modely založené na údajoch ukazujú postupnosť akcií zapojených do spracovania vstupných údajov a generovania súvisiaceho výstupu - dataflow.
- Sú obzvlášť užitočné pri analýze požiadaviek, pretože sa dajú použiť na zobrazenie komplexného spracovania v systéme.

Model činnosti činnosti inzulínovej pumpy



Spracovanie objednávky



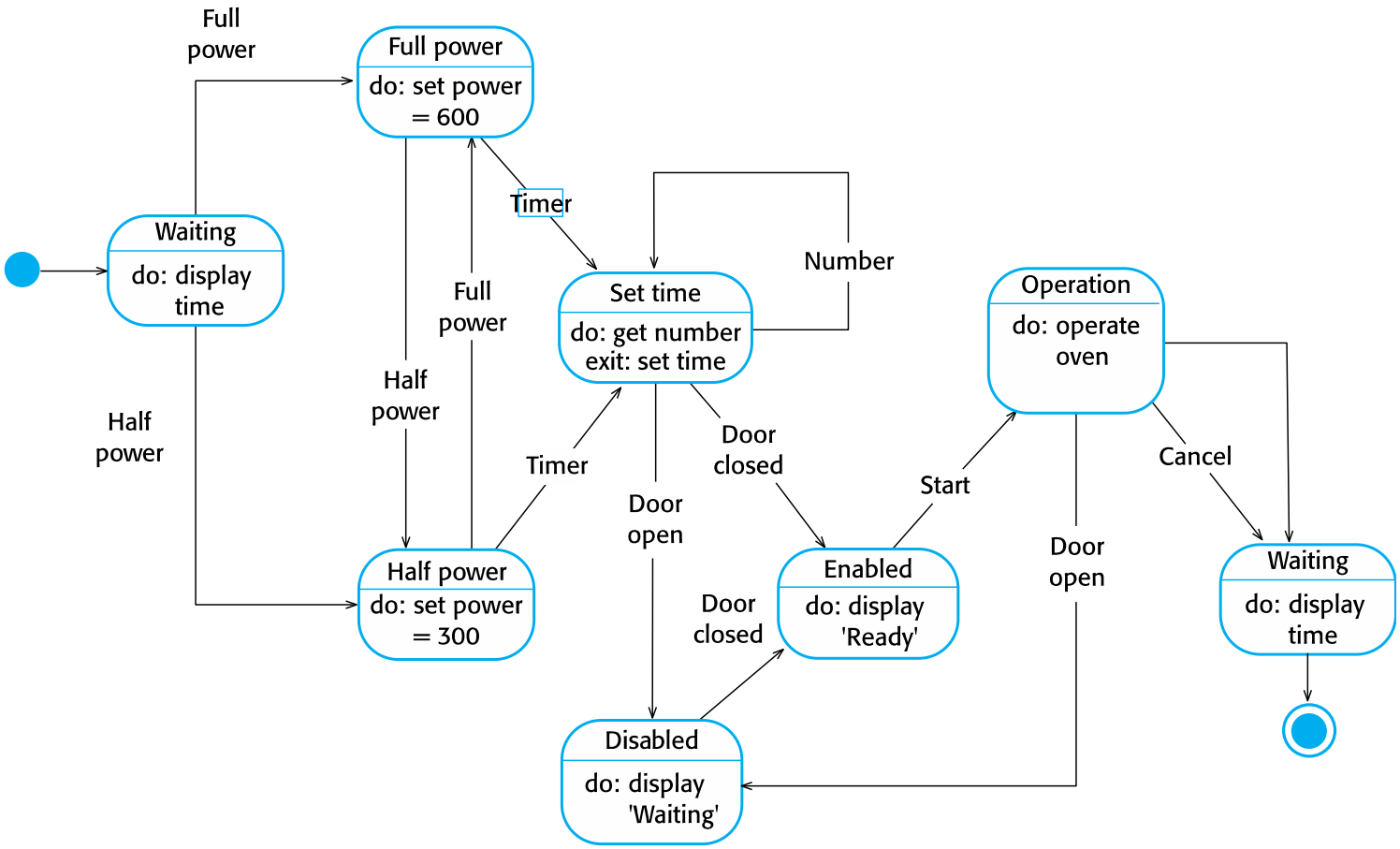
Modelovanie riadené udalosťami

- Systémy v reálnom čase sú často riadené udalosťami s minimálnym spracovaním údajov. Napríklad systém prepínania pevnej linky reaguje na udalosti, ako je „zdvihnutie sluchátka“ generovaním oznamovacieho tónu.
- Modelovanie riadené udalosťami ukazuje, ako systém reaguje na vonkajšie a vnútorné udalosti.
- Vychádza z predpokladu, že systém má konečný počet stavov a že udalosti (podnety) môžu spôsobiť prechod z jedného stavu do druhého.

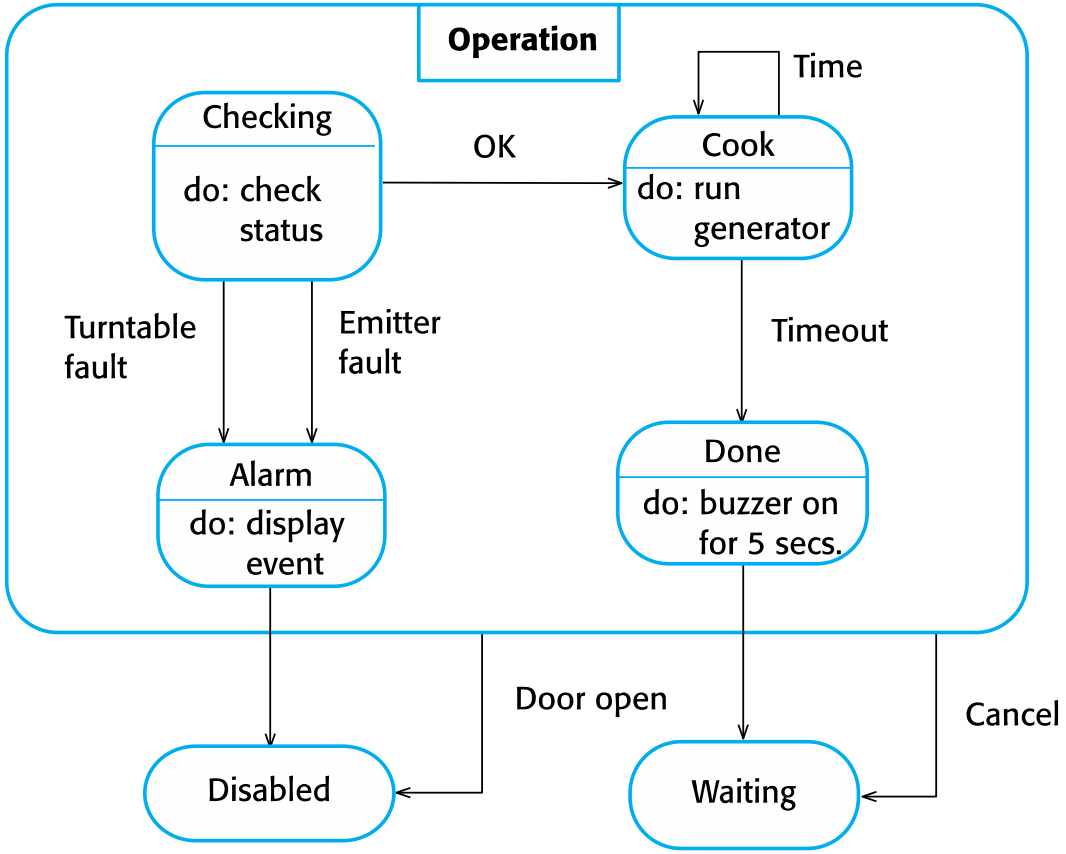
Modely stavových strojov

- Tieto modelujú správanie systému v reakcii na vonkajšie a vnútorné udalosti.
- Zobrazujú reakcie systému na podnety, takže sa často používajú na modelovanie systémov v reálnom čase.
- Modely stavových strojov zobrazujú stavy systému ako uzly a udalosti ako oblúky medzi týmito uzlami. Keď dôjde k udalosti, systém sa presunie z jedného stavu do druhého.
- Stavové diagramy sú neoddeliteľnou súčasťou UML a používajú sa na reprezentáciu modelov stavových strojov.

Stavový diagram mikrovlnnej rúry



Stav "Prevádzka" mikrovlnnej rúry



Stavy pre mikrovlnnú rúru

Štát	Popis
Čakanie	Rúra čaká na vstup. Na displeji sa zobrazí aktuálny čas.
Polovičný výkon	Výkon rúry je nastavený na 300 wattov. Na displeji sa zobrazí „Polovičný výkon“.
Plný výkon	Výkon rúry je nastavený na 600 wattov. Na displeji sa zobrazí „Plný výkon“.
Nastav čas	Čas varenia je nastavený na hodnotu zadanú používateľom. Displej zobrazuje zvolený čas varenia a aktualizuje sa podľa nastavenia času.
Zakázané	Prevádzka rúry je z bezpečnostných dôvodov vypnutá. Vnútorné osvetlenie rúry je zapnuté. Displej zobrazuje 'Nie je pripravený'.
Povolené	Prevádzka rúry je aktivovaná. Vnútorné osvetlenie rúry je vypnuté. Displej zobrazuje „Pripravené na varenie“.
Prevádzka	Rúra v prevádzke. Vnútorné osvetlenie rúry je zapnuté. Displej zobrazuje odpočítavanie časovača. Po dokončení varenia zaznie na päť sekúnd bzučiak. Osvetlenie rúry je zapnuté. Displej zobrazuje 'Varenie je dokončené', zatiaľ čo znie bzučiak .

Prechody pre mikrovlnnú rúru

Stimulácia	Popis
Polovičný výkon	Používateľ stlačil tlačidlo polovičného napájania.
Plný výkon	Používateľ stlačil tlačidlo plného napájania.
Časovač	Používateľ stlačil jedno z tlačidiel časovača.
číslo	Používateľ stlačil číselnú klávesu.
Dvere otvorené	Spínač dvierok rúry nie je zatvorený.
Dvere zatvorené	Spínač dvierok rúry je zatvorený.
Štart	Používateľ stlačil tlačidlo Štart.
Zrušiť	Používateľ stlačil tlačidlo Zrušiť .

Modelom riadené inžinierstvo

- Modelom riadené inžinierstvo (MDE) je prístup k vývoju softvéru, kde hlavnými výstupmi vývojového procesu sú skôr modely ako programy.
- Programy, ktoré sa spúšťajú na hardvérovej/softvérovej platforme, sa potom generujú automaticky z modelov.
- Zástancovia MDE tvrdia, že to zvyšuje úroveň abstrakcie v softvérovom inžinierstve, takže inžinieri sa už nemusia zaoberať detailmi programovacieho jazyka alebo špecifikami vykonávacích platforiem.

Použitie modelom riadeného inžinierstva

- Modelom riadené inžinierstvo je stále v ranom štádiu vývoja a nie je jasné, či bude mať významný vplyv na prax softvérového inžinierstva alebo nie.
- Výhody
  - Umožňuje posúdiť systémy na vyšších úrovniach abstrakcie
  - Automatické generovanie kódu znamená, že je lacnejšie prispôsobiť systémy novým platformám.
- Nevýhody
  - Modely pre abstrakciu nie sú úplne vhodné na implementáciu.
  - Úspory z generovania kódu môžu byť vyvážené nákladmi na vývoj prekladačov pre nové platformy.

Architektúra riadená modelom

- Modelom riadená architektúra (MDA) bola predchodcom všeobecnejšieho modelom riadeného inžinierstva
- MDA je modelovo zameraný prístup k návrhu a implementácii softvéru, ktorý využíva podmnožinu modelov UML na popis systému.
- Vytvárajú sa modely na rôznych úrovniach abstrakcie. Z vysokoúrovňového, platformovo nezávislého modelu je v princípe možné vygenerovať pracovný program bez manuálneho zásahu.

Typy modelu

- ✧ Model nezávislý na výpočte (CIM)
  - Tieto modelujú dôležité doménové abstrakcie používané v systéme. CIM sa niekedy nazývajú doménové modely.
- ✧ Model nezávislý na platforme (PIM)
  - Tieto modelujú fungovanie systému bez odkazu na jeho implementáciu. PIM je zvyčajne opísaný pomocou modelov UML, ktoré ukazujú statickú štruktúru systému a ako reaguje na vonkajšie a vnútorné udalosti.
- ✧ Modely špecifické pre platformu (PSM)
  - Ide o transformácie modelu nezávislého na platforme so samostatným PSM pre každú aplikačnú platformu. V zásade môžu existovať vrstvy PSM, pričom každá vrstva pridáva určité detaily špecifické pre platformu.

Agilné metódy a MDA

- Vývojári MDA tvrdia, že je určený na podporu iteratívneho prístupu k vývoju, a preto ho možno použiť v rámci agilných metód.
- Pojem rozsiahleho dopredu vykonaného modelovania je v rozpore so základnými myšlienkami v agilnom manifeste a je podozrenie, že len málo agilných vývojárov sa stotožňuje s modelom riadeným inžinierstvom.

Prijatie MDA

- Prijatie MDE/MDA obmedzuje celý rad faktorov
- Na konverziu modelov z jednej úrovne na druhú je potrebná podpora špeciálnych nástrojov
- Dostupnosť nástrojov je obmedzená a organizácie môžu vyžadovať prispôsobenie nástrojov a prispôsobenie ich prostrediu
- Pokiaľ ide o systémy s dlhou životnosťou vyvinuté pomocou MDA, spoločnosti sa zdráhajú vyvíjať svoje vlastné nástroje alebo sa spoliehať na malé spoločnosti, ktoré môžu skončiť s podnikaním.

Prijatie MDA

- Modely sú dobrým spôsobom, ako uľahčiť diskusie o návrhu softvéru. Abstrakcie, ktoré sú užitočné pre diskusie, však nemusia byť tými správnymi abstrakciami na implementáciu.
- V prípade najkomplexnejších systémov nie je implementácia hlavným problémom – dôležitejšie sú inžinierstvo požiadaviek, bezpečnosť a spoľahlivosť, integrácia so staršími systémami a testovanie.

Prijatie MDA

- Argumenty pre platformovú nezávislosť platia len pre veľké systémy s dlhou životnosťou. V prípade softvérových produktov a informačných systémov budú úspory vyplývajúce z používania MDA pravdepodobne prevážené nákladmi na jeho zavedenie a nástroje.
- Široké prijímanie agilných metód v rovnakom období, v ktorom sa MDA vyvíjalo, odviedlo pozornosť od prístupov založených na modeloch.
- V praxi sa teda najčastejšie stretnete s týmito (v poradí):
  - Model-driven analysis
  - Model-driven architecture
  - Model-driven engineering