Grafický popis algoritmu

Vývojový diagram

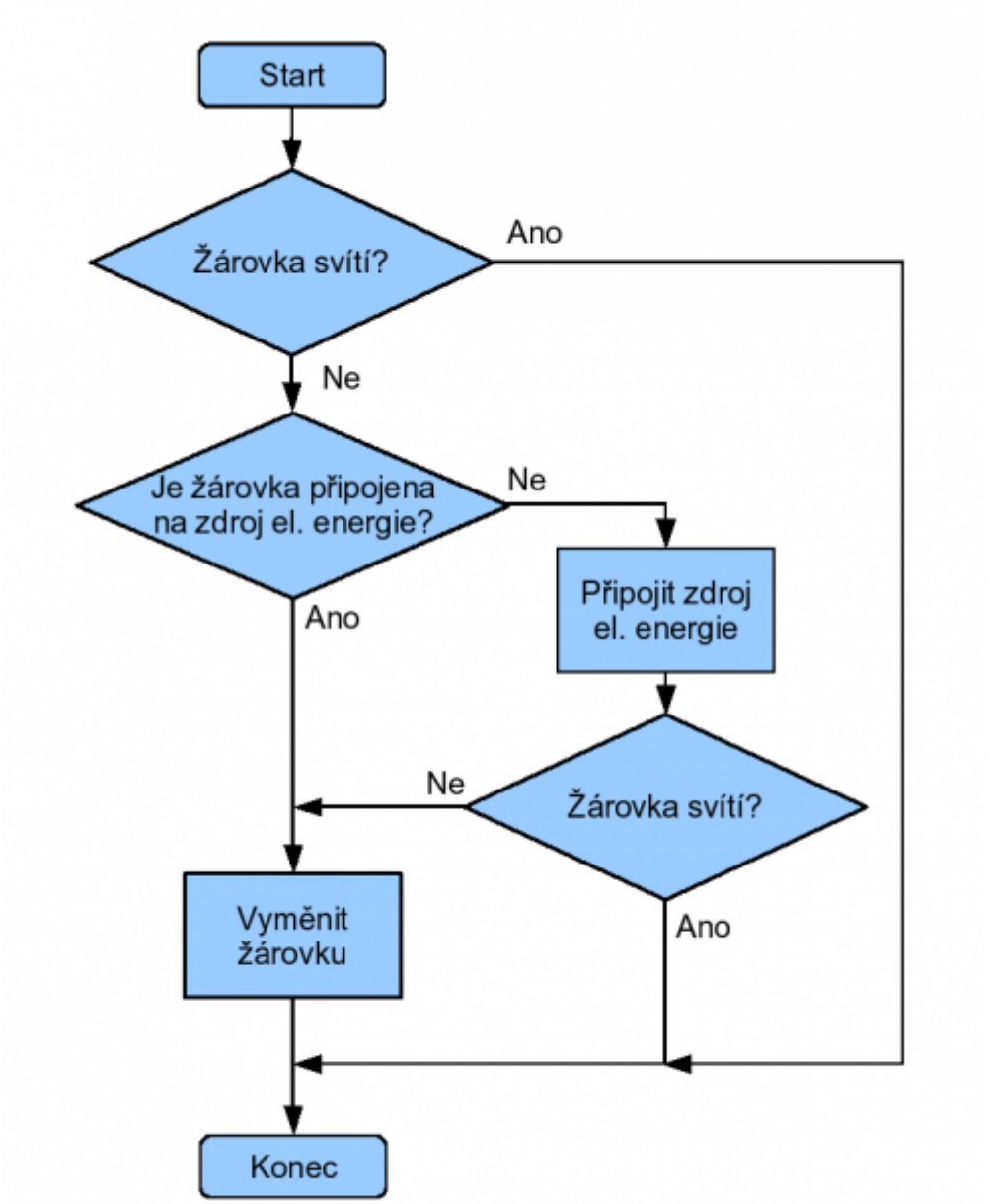
Vývojový diagram slouží ke grafickému znázornění algoritmu pomocí symbolů a čar (šipek). Algoritmus je přesný postup, kterým lze vyřešit danou úlohu.

Pravidla

Ve vývojovém diagramu se postupuje shora dolů a zleva doprava, lze však změnit směr užitím šipek.

Pro zápis diagramu používáme následující symboly (spolu s vnitřními popisy):

- **obdélník** – krok algoritmu (příkazy – například i výpis)
- **kosočtverec** – větvení postupu algoritmu podle naplnění podmínky; bývá (z hlediska ISO normy pro neimperativní paradigmata nesprávně) používán i jako podmínka pro cykly
- **zaoblený obdélník** – počátek nebo ukončení algoritmu
- **obdélník se svislými čarami po stranách** – dodatečné podprogramy
- **rovnoběžník** – vstup
- **obdélník s ořezanými rohy** – ohraničení kroků cyklu tak, aby byly ohraničeny neořezanými stranami (pro *for* a *while* je podmínka uvnitř horní hranice, pro *do while* uvnitř hranice spodní)



Druhy vývojových diagramů

- **document flowcharts** – řízení toků dokumentů
- **data flowcharts** – řízení toků dat
- **system flowcharts** – řízení toků fyzické vrstvy nebo vrstvy zdrojů
- **program flowcharts** – řízení toků v programu

UML

Název z angličtiny – **unified modeling language**.

Jedná se o grafický jazyk pro vizualizaci a návrhy programových způsobů. Podporuje objektově orientovaný přístup k analýze, ale již nespecifikuje metodiku funkce programu.

Účely UML

- **Kreslení konceptu** – do diagramů se nakreslí podstatné věci před programováním. Modelovací software jako Umbrello poté umožňuje generování šablony kódu podle UML konceptu.
- **Kreslení detailních návrhů** – umožňují programátorovi lépe pochopit analytické zadání.
- **Jako programovací jazyk** – kód spustitelný přímo z diagramů, v této souvislosti se často používá pojem MDA (model driven architecture).

Základní dělení diagramů UML

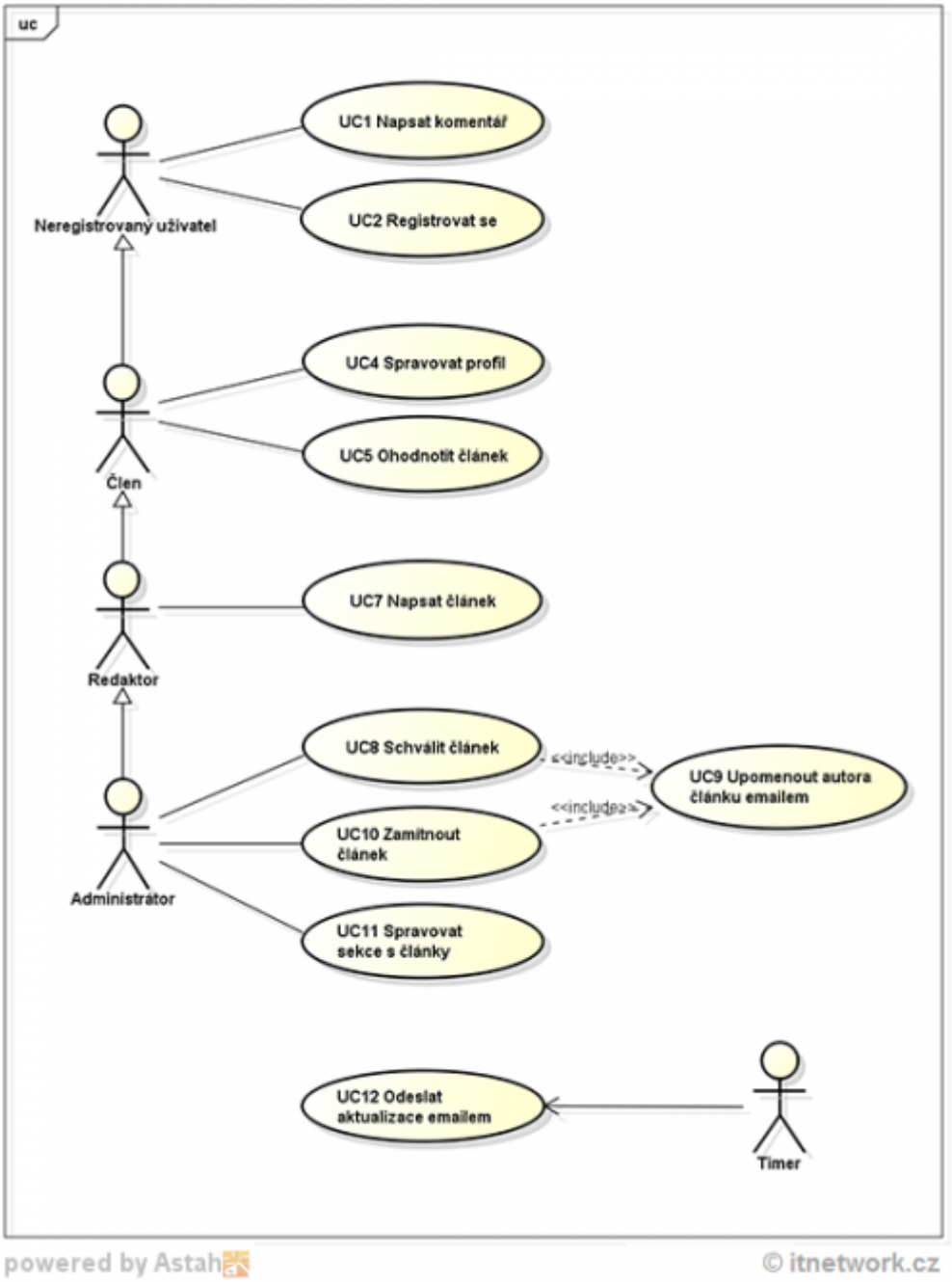
UML umožňuje tvorbu několika typů diagramů, které můžeme kategorizovat podle toho, jaké aspekty systému reprezentují:

- **Diagramy použití a chování** – zobrazují vnější chování systému a jeho funkce (use case diagramy).
- **Diagramy struktur** – zobrazují prvky, které musí být přítomny v systému, aby mohl správně fungovat, respektive jeho logickou strukturu (diagramy tříd, diagramy objektů, diagramy komponent).
- **Diagramy interakcí** – speciální druh diagramů chování, který se zaměřuje na znázornění toku dat a komunikace uvnitř modelovaného systému (sekvenční diagramy, komunikační diagramy).

Use case diagramy

Doslova diagramy „případů užití“ – zobrazují funkcionalitu systému – pouze chování programu (nikoliv přesnou realizaci), a to přímo očima koncového uživatele. Diagram tedy pouze ukazuje, co má systém umět.

Základní dva prvky diagramu jsou značky aktérů (actors – uživatelé) a případů užití (use cases – jednotlivé možnosti, které se aktérům nabízejí).



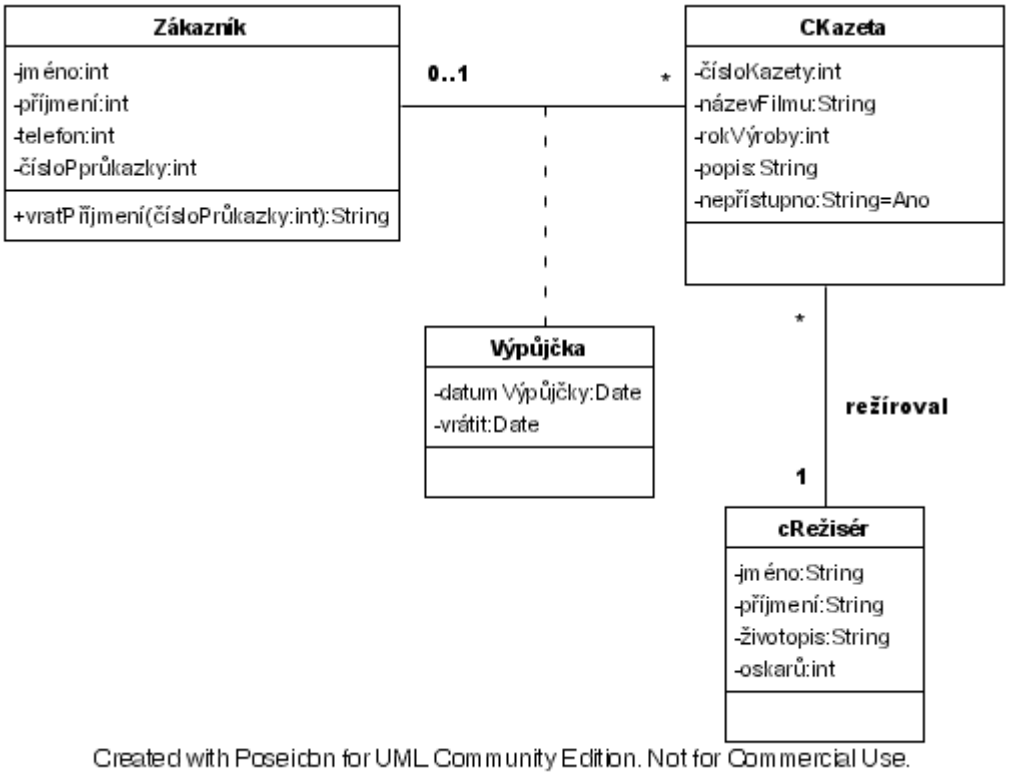
Diagramy tříd

Diagramy tříd se využívají k zobrazení tříd, jejich metod, atributů a vztahů mezi nimi.

Vztahy a značení

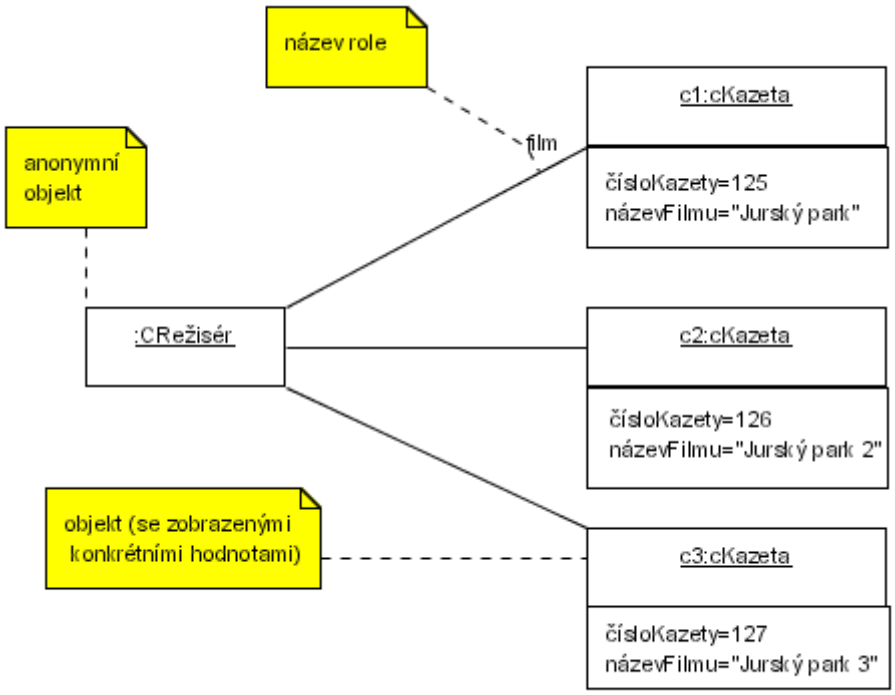
- **Asociace** – vztah informuje o spojení instancí dvou tříd
- **Agregace** – vztah celek – část
- **Kompozice** – silnější verze agregace
- **Dědičnost** – potomek dědí atributy předka
- **Závislost** – změna jedné třídy ovlivní třídu druhou
- **Realizace** – souhrn všech veřejně dostupných metod dané třídy
- **+** = public

- **−** = private
- **#** = protected



Diagramy objektů

Diagramy objektů (nebo také instancí) fungují podobně jako diagramy tříd, ovšem zachycují stav běžícího systému v přesně daném momentu – nezobrazují třídy, ale jejich konkrétní instance, u kterých očekáváme, že by se v systému mohly za běhu vyskytnout.



Sekvenční diagramy

Sekvenční diagramy (také diagramy událostí) se starají o znázornění posloupného zasílání zpráv a požadavků mezi sledovanými objekty. Ty jsou znázorněny v horní části diagramu a vychází z nich svislé přerušované čáry (lifelines) indikující jejich existenci (při smazání končí znakem X). Následně se (zleva) posílají objektům požadavky pomocí šipek doplněných o název volané funkce. Šipky mohou být rovněž přerušované, v takovém případě však značí odpovědi na požadavky. Doba zpracování požadavků se znázorňuje obdélníky na lifeline.