Návrhové vzory datové vrstvy

Kašpar, Révay, Šmíd

Obecné úvahy při návrhu datové vrstvy

- Mapování aplikačních entit na struktury datových zdrojů
- Odstínění přístupu k datům od business logiky
- Správa spojení
- Škálovatelnost a výkon
- Bezpečnostní rizika

Table Data Gateway (TDG)

- Často označován také jako Data Access Object (DAO)
- Veškerou logiku spojenou se spravováním záznamů jedné tabulky v databázi vkládá do jedné třídy
  - Jeden objekt (instance) pro každou jednu tabulku/pohled
  - CRUD operace (každá metoda mapuje vstupy na SQL příkazy, které se poté spouští nad DB)
  - Objekt je bezestavový
  - Může jako výsledek operace vrátit jeden nebo více záznamů z tabulky
  - Získané záznamy z DB může vracet např. jako Data Transfer Object (DTO) nebo přímo doménový objekt (při použití Doménového modelu -> obousměrná závislost mezi doménovým objektem a gateway (úzce propojené))
- Vhodné použít například ve spojení s Transaction Scripts, naopak ve spojení s Domain Model je lepší použít Data Mapper, který poskytuje lepší izolaci než TDG
- Lepší použít v momentě, kdy chceme manipulovat s množinou dat, než-li jen s jedním záznamem

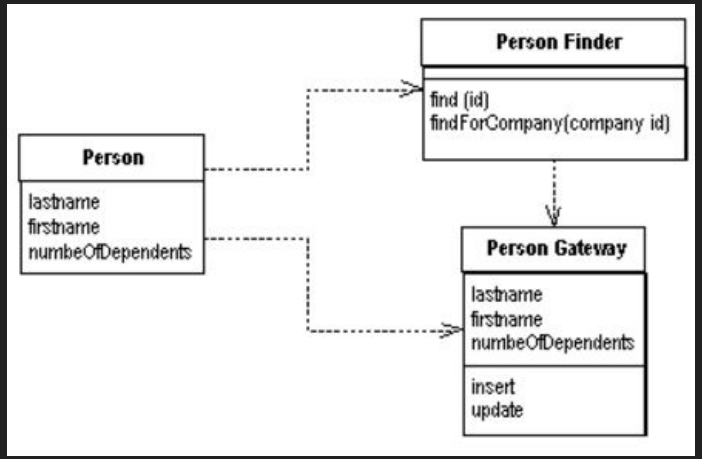
Table Data Gateway (TDG)

- Výhody:
  - Abstrakce (izolace) přístupu business logiky od přímého přístupu k datům v úložišti
    - změna business logiky neovlivní datovou vrstvu a naopak
    - Lepší organizace kódu - všechny SQL příkazy na jednom místě -> lepší údržba kódu a systému, přehlednost…
  - Jeden z nejjednoduších vzorů
- Nevýhody:
  - Potenciálně abstrakce
    - může být nutné volat více metod k získání určitých dat, které by ve skutečnosti bylo možné získat jedním SQL příkazem

Row Data Gateway (RDG)

- Podobně jako TDG představuje jistou abstraktní vrstvu mezi business logikou a datovým úložištěm
  - všechny detaily o datovém zdroji jsou schovány za tímto rozhraním
- RDG objekt odpovídá jednomu záznamu tabulky
  - jedna instance na záznam
  - je stavový - atributy odpovídají všem sloupcům tabulky
- Find metody mohou být buď statické, nebo v separátní Finder třídě
  - každá tabulka bude mít jeden finder objekt pro získání záznamu a jeden gataway objekt pro výsledky
- Je vhodné použít opět ve spojení s Transaction Scripts vzorem
- Použijeme v momentě, kdy chceme pracovat s daty ze dvou či tří záznamů tabulky najednou, ale nechceme pro ně vytvářet nový extra objekt
- Podobný Active Record vzoru, ale narozdíl od něj neobsahuje žádnou (doménovou) logiku, stará se pouze o zpřístupňování dat

Row Data Gateway



- Výhody:
  - odstínění business logiky od datového zdroje
  - použitelný jak pro tabulky, tak i pohledy
- Nevýhody:
  - Při použití tohoto vzoru je třeba být opatrný - pokud máme dva gateway objekty pracující nad stejnými tabulkami, může se stát, že aktualizace hodnot v jednom gateway zruší změny ve druhém (vzájemné ovlivňování hodnot)

Active Record Pattern

- Objekt který představuje řádek v databázi nebo pohled do databáze
- K informacím z databáze přidává aplikační logiku

Výhody

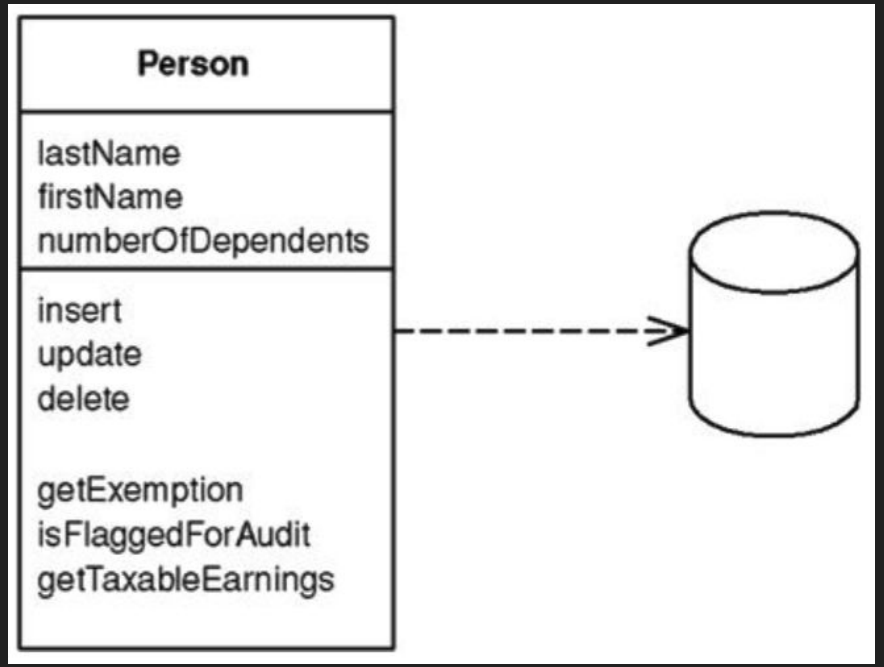
- Vhodný v případě, že doménová logika není moc komplexní (CRUD, validace, ...)
- Jednoduchost

Nevýhody

- Objektový a databázový návrh je svázaný

Active Record Pattern

- Vytvoření Active Record instance z výsledku (řádku) SQL dotazu
- Vytvoření nové instance Active Record
- Statické metody pro vyhledávání
- Update dat v databázi
- Get a Set metody pro atributy
- Metody business logiky



Data Mapper

- Slouží pro tok dat mezi pamětí (in-memory objects) a persistentní vrstvou
- Obshauje CRUD operace pro daný entitní typ
- Obsahuje mapu identit - reference na již načtené objekty
  - Snižuje zátěž na databázi
  - Zvyšuje performance aplikace
- Převádí strukturu persistentních dat na objekty a naopak
- Umožňuje “inteligentně” načítat data z persistentní vrstvy
  - Načtení objektu může vyvolat načítání i některých asociovaných objektů
  - Nutno při návrhu mapperu mít znalost o tom jak jsou objekty v aplikaci používány
- Vhodný pro aplikace s komplexní business logikou
- Nevhodný pro malé aplikace s jednoduchou logikou

ORM (Object Relational Mapping)

- Mapování objektů do relační databáze

Řeší následující problémy (Object-relational impedance mismatch):

- Granularita
- Dědičnost
- Identita
- Asociace
- Navigace v datech

ORM (Object Relational Mapping)

Výhody

- Frameworky řeší většinu práce

Nevýhody

- Použitím frameworku ztrácí vývojář kontrolu nad SQL dotazy
- Pro dosažení dobrého výkonu je potřeba znalost frameworku

Repository

- Představuje abstrakci, jehož cílem je redukovat celkovou složitost a zajistit, že se zbytek kódu nemusí starat o ukládání dat (detaily, technologie a veškerá logika v pozadí)
  - leží mezi business a datovou vrstvou (vystupuje v roli in-memory kolekce)
  - získává data z dat. zdroje, která mapuje na business entity (entity doménového modelu) a změny provedené v business entitě ukládá do dat. zdroje
  - centralizace logiky datového přístupu nebo přístupu k web. službám
  - zlepšení testovatelnosti kódu
- Poskytuje flexibilní architekturu, která je schopná se přizpůsobit změnám spojeným s vývojem aplikace
- Může využívat Data Mapper pro převod mezi business entitou a entitou dat. zdroje
- Umožňuje dotazování dat, přidání nebo odebrání do/z datového zdroje na podobné bázi jako v případě použití kolekcí dat
- Vhodný v případech, kdy je v systému velké množství doménových tříd nebo složité dotazovací příkazy
- Při návrhu postupovat směrem od kódu k databázi (prvně se vybudují třídy v doménovém modelu) nikoli naopak (vytvoříme třídy podle toho jaké máme tabulky v databázi)

Repository

- Výhody:
  - Umožňuje vybrat vhodnou technologii (změna databáze, použití webové služby…) pro konkrétní případy užití
  - Umožňuje psát unit testy místo integračních testů (redukuje složitost testů a umožňuje psát unit testy pro business vrstvu a integrační testy pro dat. vrstvu) -> lepší testovatelnost kódu
  - snížení duplicity kódu
- Nevýhody:
  - není tak vykonný (kvůli vysoké abstrakci)

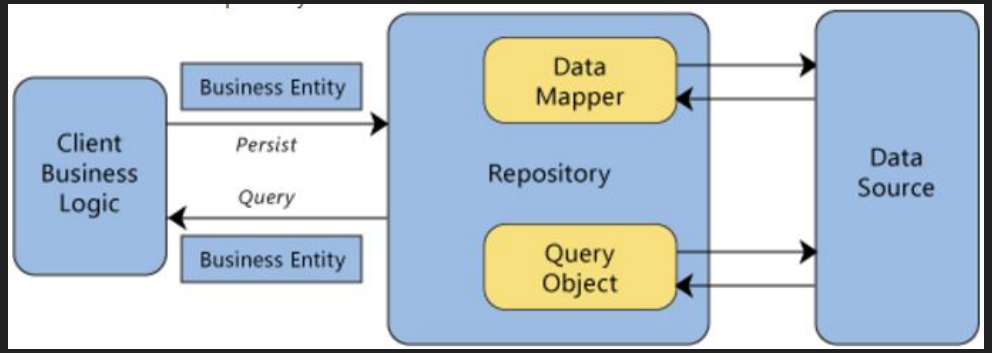


Diagram komponent

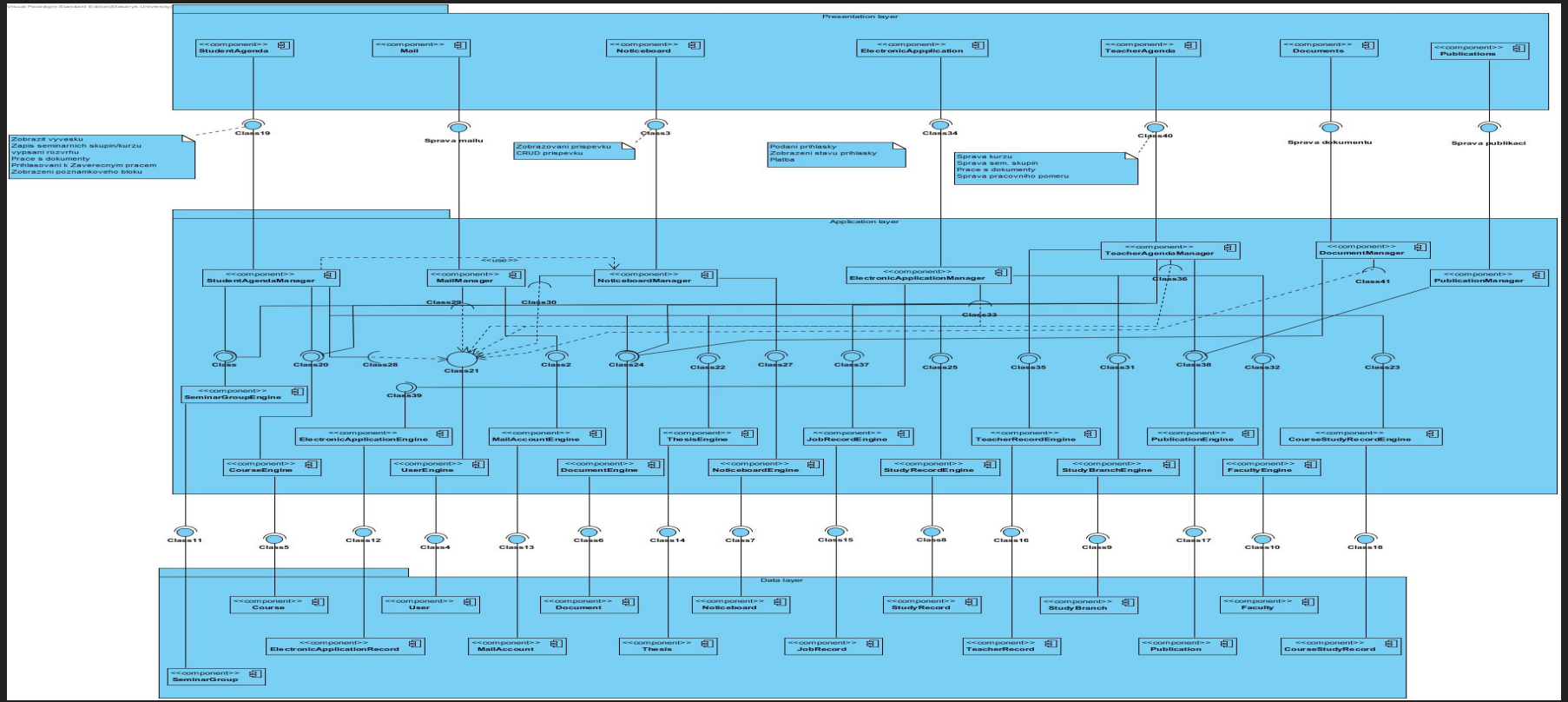
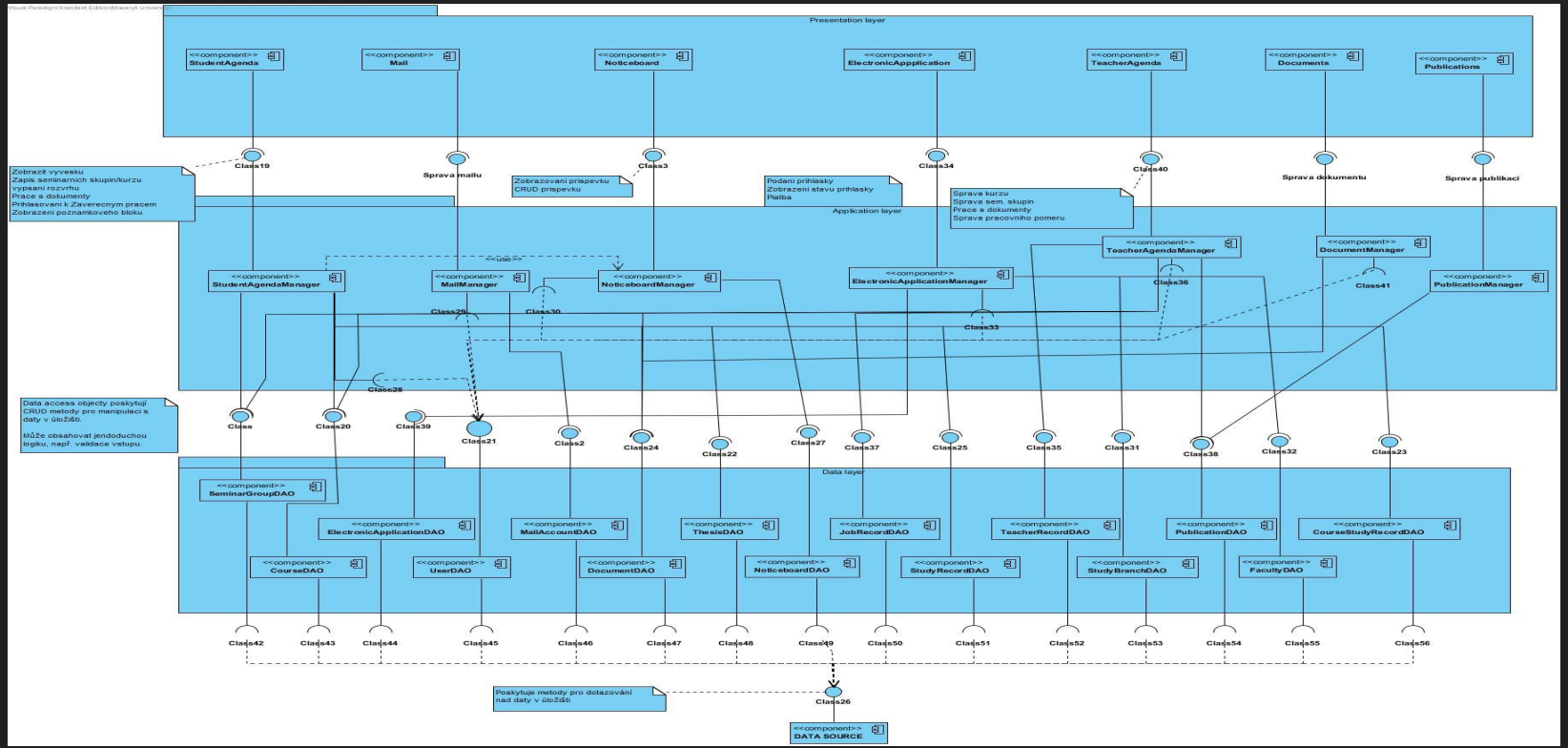


Diagram komponent s Table Data Gateway



Zdroje

1. Systems Engineering - EAA - Patterns of Enterprise Application Architecture - Addison Wesley
2. https://msdn.microsoft.com/en-us/library/ee658127.aspx#ObjectRelationalMappingConsiderations
3. http://hibernate.org/orm/what-is-an-orm/
4. https://en.wikipedia.org/wiki/Data_access_object
5. https://www.martinfowler.com/eaaCatalog/tableDataGateway.html
6. https://www.martinfowler.com/eaaCatalog/rowDataGateway.html
7. http://blog.gauffin.org/2013/01/repository-pattern-done-right/
8. https://martinfowler.com/eaaCatalog/repository.html
9. https://msdn.microsoft.com/en-us/library/ff649690.aspx