Vývoj informačních systémů

Přehled témat a úkolů

2023-24

Organizace výuky

- doc. Mgr. Miloš Kudělka, Ph.D.
  - EA 439, +420 597 325 877
  - homel.vsb.cz/~kud007
  - milos.kudelka@vsb.cz

- Přednáška
  - Znalosti
  - Schopnosti

- Cvičení
  - Diskuze
  - Prezentace
  - Poznámky k vývoji a využívání technik a technologií
  - Návrh a programování

Klíčové principy

- Porozumění tomu co, proč a jak se dělá.
- Dělat dobře znamená dělat tak, jak to úspěšně dělají druzí.
- Poznat technologie z druhé strany.

Vstupní znalosti

- Objektově orientovaný přístup
- UML
- Návrhové vzory (GoF)
  - Gamma, E., Helm, R., Johnson, R., Vlissides, J. (2003). *Návrh programů pomocí vzorů*. Grada, Praha.
  - Pecinovský, R. (2007). *Návrhové vzory: [33 vzorových postupů pro objektové programování]*.

Informační systém

- V širším slova smyslu se jedná o interakci mezi lidmi, procesy a daty. Informační systém je určen ke zpracování (získávání, přenos, uložení, vyhledávání, manipulace, zobrazení) informací.

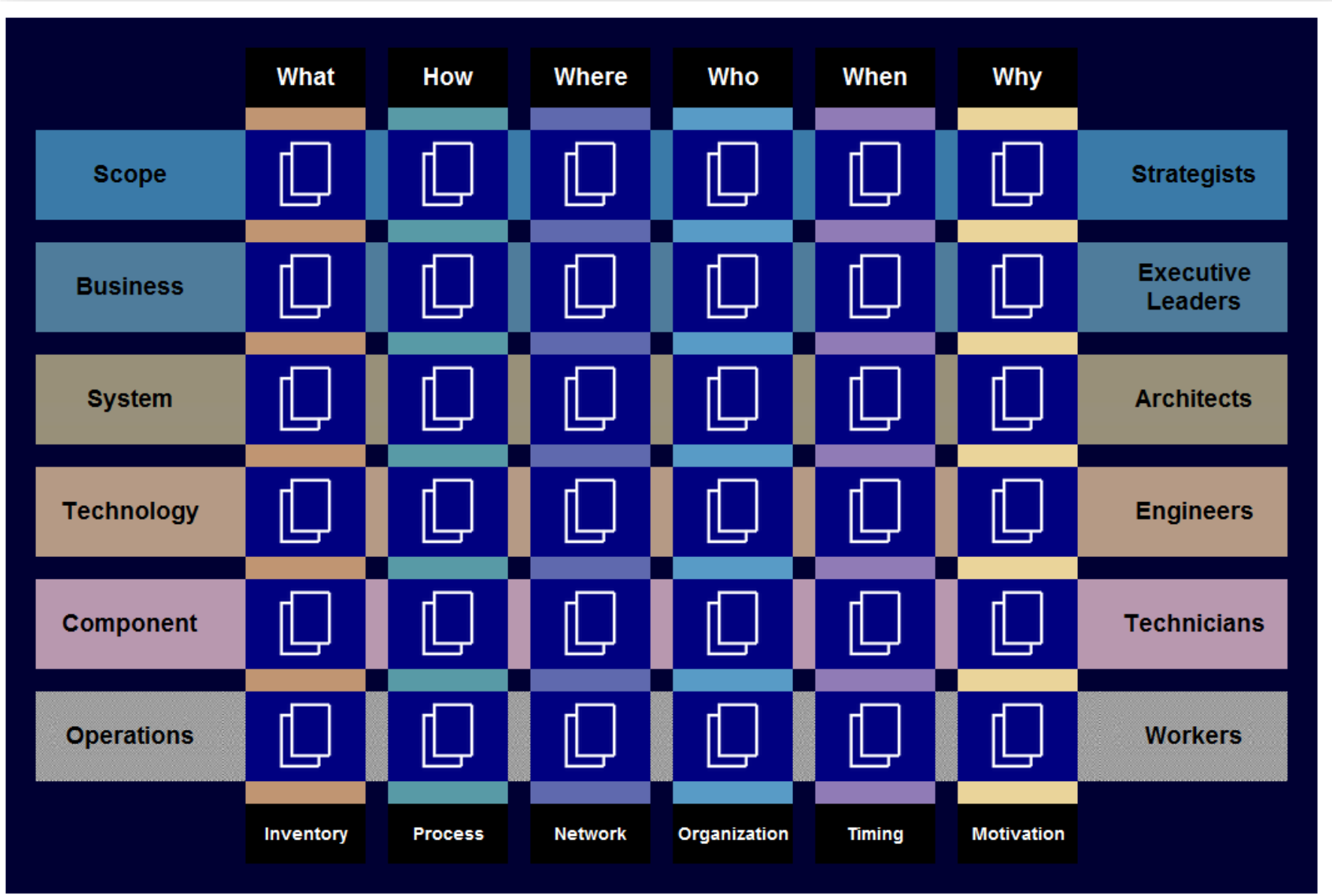
Klasifikace

- Různé typy agend
  - Ekonomická
  - Personální
  - Skladová
  - Dokumentová (např. spisová služba)
  - Školní

- ERP, CRM, CMS, DMS, Project management…

Architektura

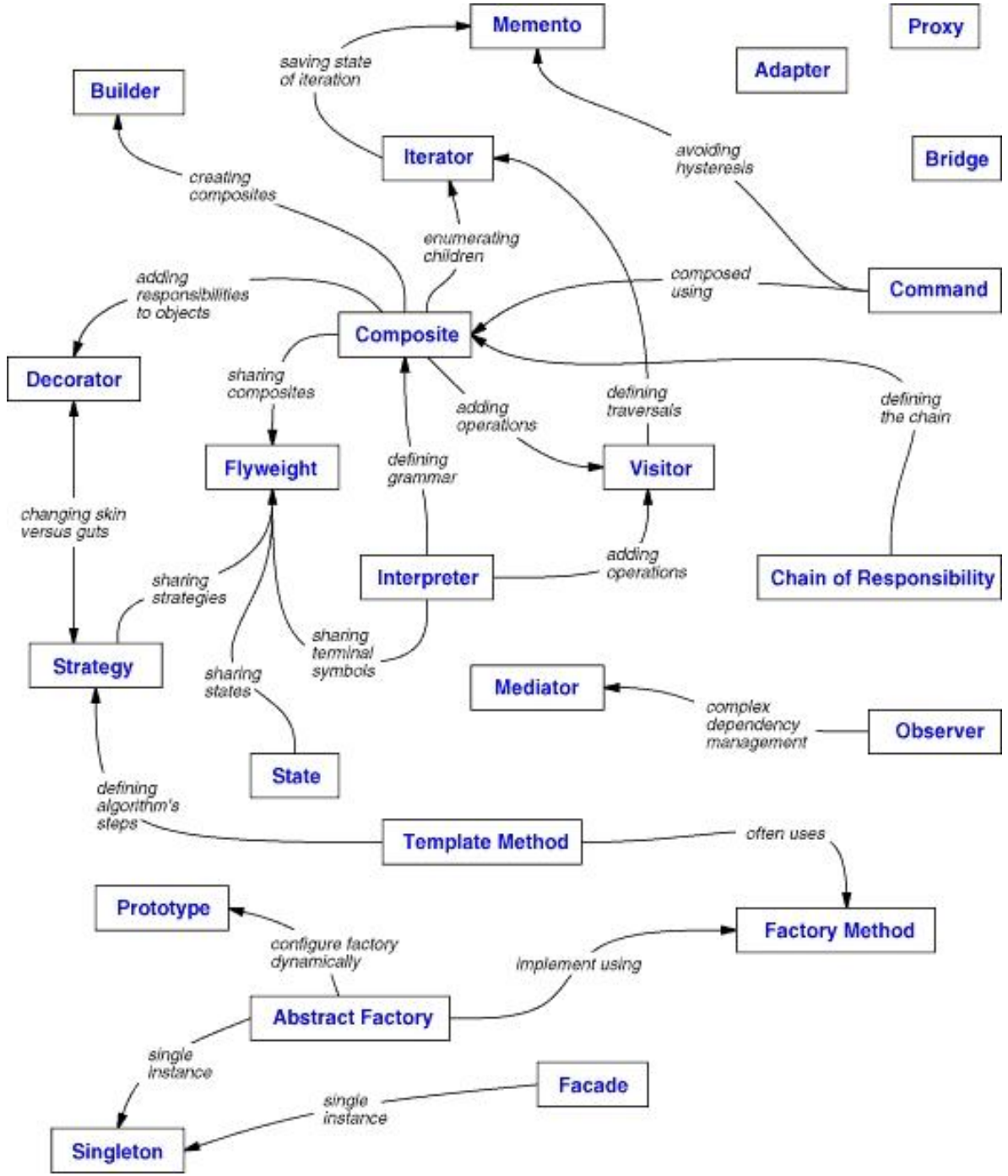
- Architektura informačního systému leží na vyšší úrovni abstrakce tak, že zahrnuje
  - pohled na **doménu** informačního systému (skupina souvisejících „věcí“ z pohledu zákazníka),
  - pohled vývojáře na globální strukturu systému a chování jeho částí, jejich propojení a synchronizace,
  - pohled na přístup k datům a toky dat v systému,
  - fyzické rozmístění komponent
  - …


What
How
Where
Who
When
Why
Scope
Business
System
Technology
Component
Operations
Strategists
Executive Leaders
Architects
Engineers
Technicians
Workers
Inventory
Process
Network
Organization
Timing
Motivation

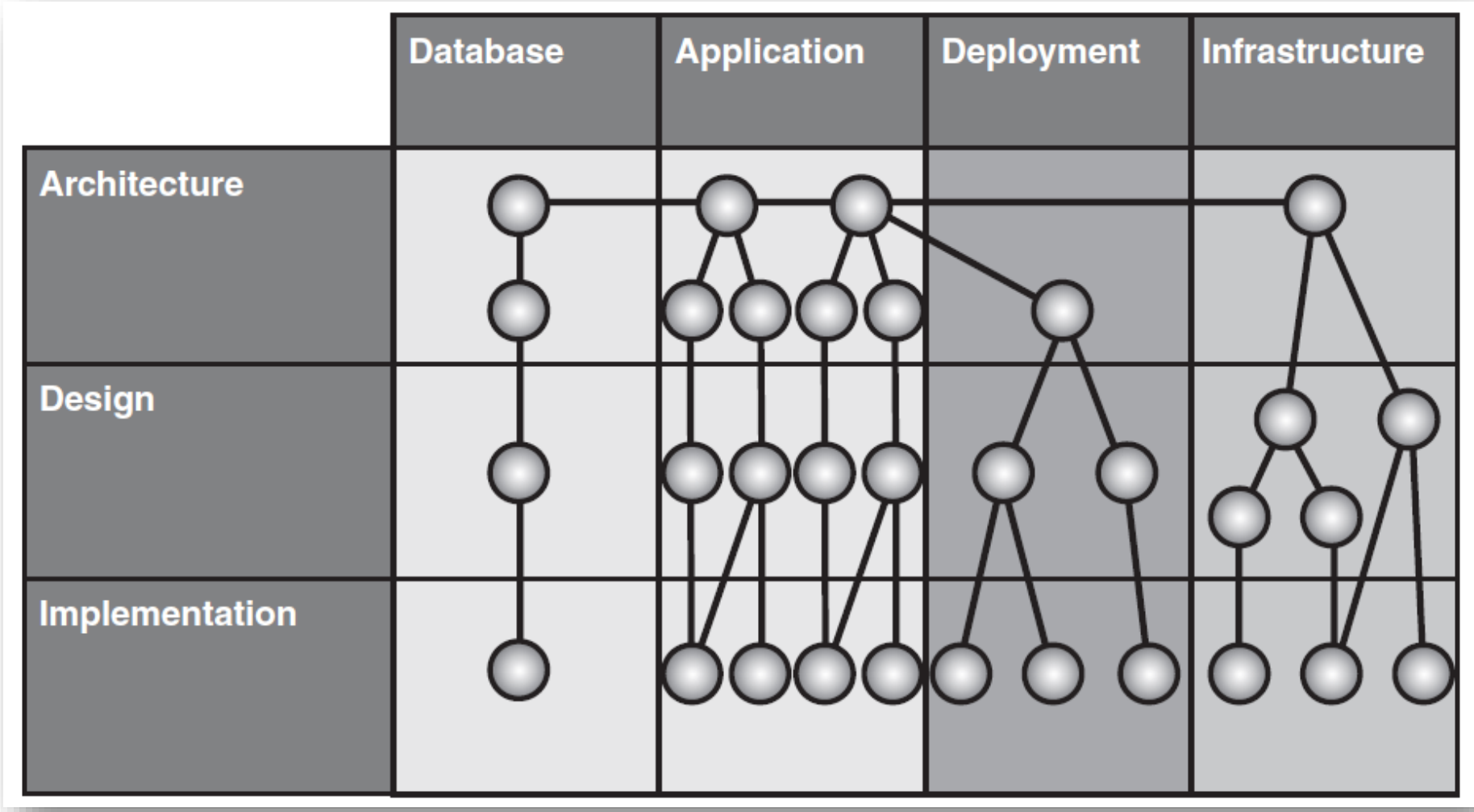
Vzory jako připravené návody

- GoF (vytváření, struktury, chování).
- Vzory a jazyky vzorů.
- Vzor je to, co opakovaně funguje.
- *Existují i antivzory (opakovaně nefungují).*

GOF: Vztahy mezi vzory – jazyk vzorů



Pohled architekta a vývojáře (abstrakce)



Three-Layered Application

Context

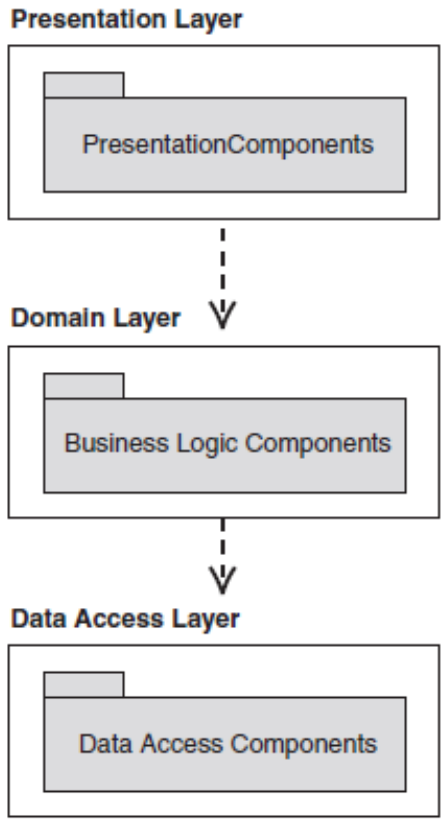
You are building a business solution using layers to organize your application

Problem

How do you organize your application to reuse business logic, provide deployment flexibility and conserve valuable resource connections?

Solution

Create three layers: presentation, business logic (domain), and data access. Place all components responsible for the view in the presentation layer. Encapsulate all business logic in domain layer components that implement well-known component interfaces. Locate all database-related code, including database client access and utility components, in the data access layer. Require the data access layer to be responsible for connection pooling when accessing resources. Make sure you eliminate the dependencies between data access components and business layer components. Either eliminate dependencies between the business layer and the presentation layer or manage the dependencies here using the *Observer* pattern.



Three-Layered Services Application

Context

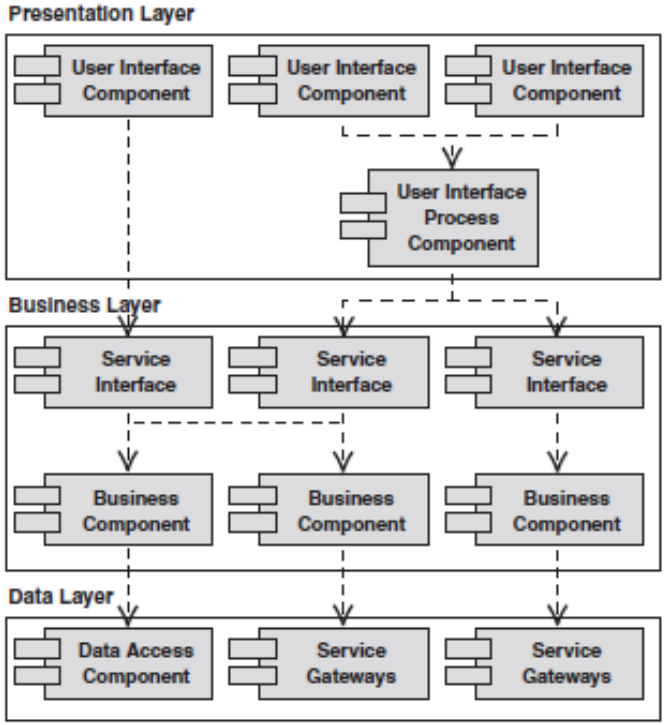
You are building a business solution that uses presentation, business, and data access layers to organize your application. You want to expose some of the core functionality of your application as services that other applications can consume and enable your application to consume other services.

Problem

How do you organize your application to provide and consume granular data and logical elements from highly variable sources?

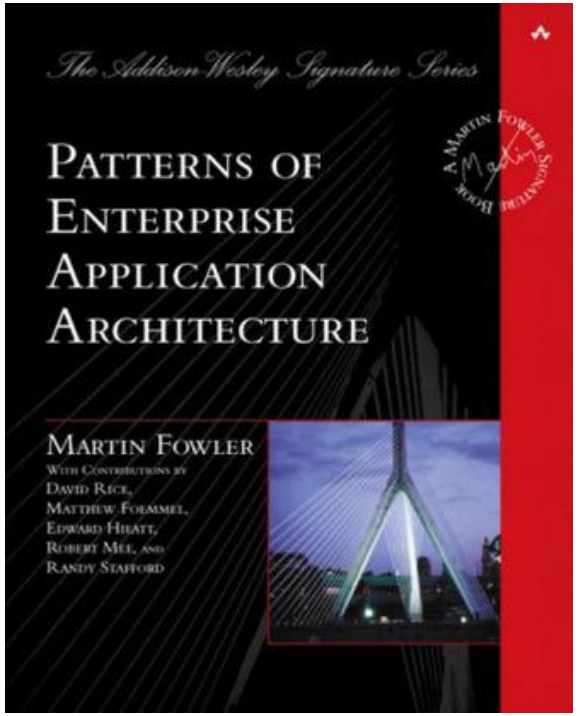
Solution

Decompose your application logic into a collaborating set of services that provide parts of the overall system functionality. Next, in the domain layer, identify a Service Interface for each service that is independent of the underlying implementation. Finally, extend the data access layer to use Service Gateways to communicate with other service providers. If application navigation logic is sufficiently complex, consider user interface process components as part of the presentation layer to encapsulate and reuse this logic.

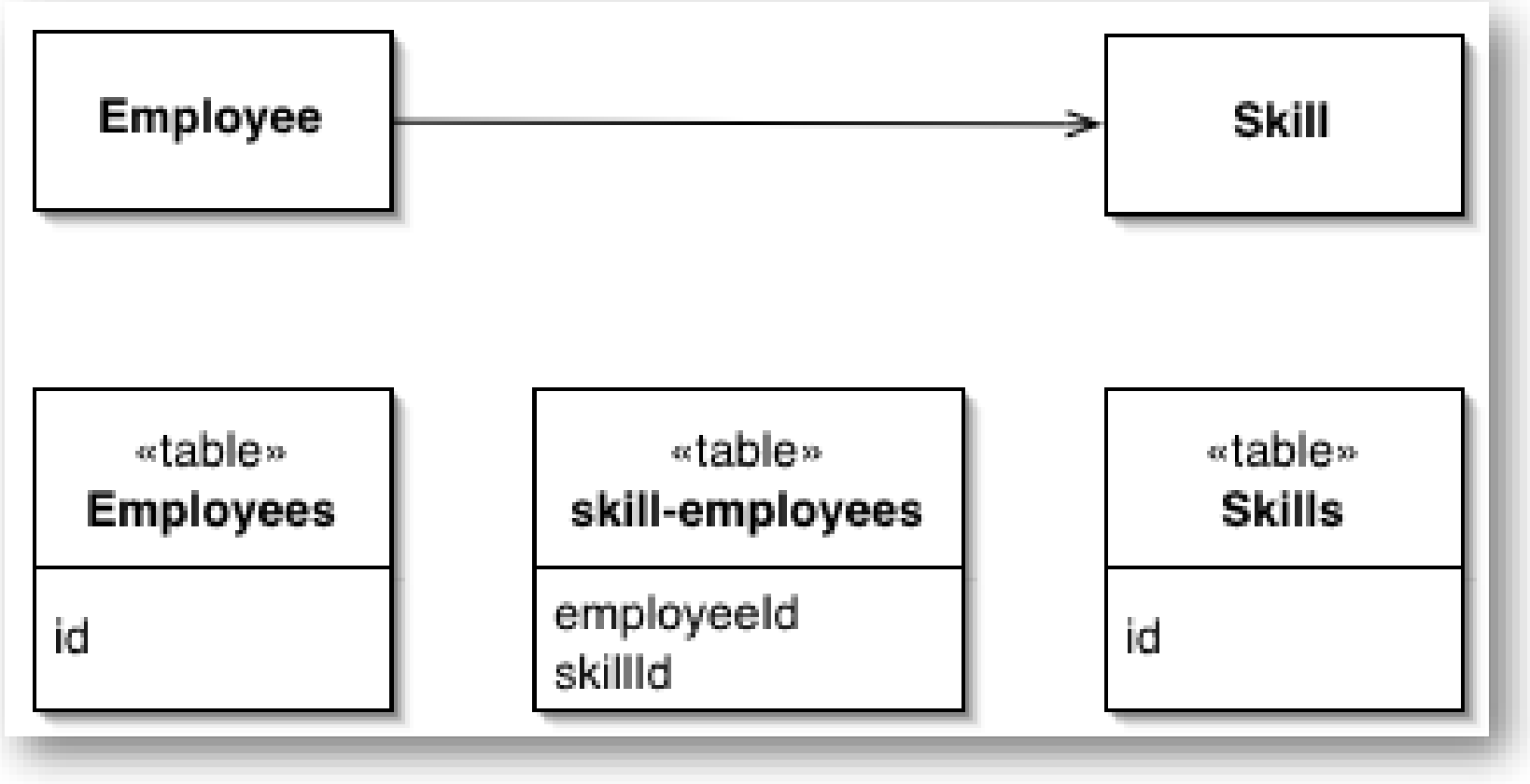


Vzory pro „enterprise“ architekturu

- Martin Fowler, Microsoft
- Rozložení aplikační architektury do více vrstev.
- Základní principy propojení doménové logiky a relačních dat.
- Principy návrhu prezentační vrstvy.



Association Table Mapping



Framework

- Java, .NET Framework,…
- Jak souvisí frameworky a vzory?
- Vzor se ze své definice implementuje vždy znovu.
- Frameworky poskytují řešení postavené na vzorech (pokud ne, něco je špatně).

Domain Specific Language

- *Domain-specific language*: Počítačový jazyk s omezenou vyjadřovací schopností.
- Zaměření na určitou konkrétní a „malou“ doménu.
- Tomu přizpůsobená povaha jazyka (programovací x deklarativní)
- Omezená vyjadřovací schopnost jazyka přizpůsobená doméně (a tedy i uživatelům z této domény).

Životní cyklus informačního systému

- Staré x nové přístupy
- Požadavky, parametry
- Přírůstky a iterace
- Kdy se začíná a kdy se končí
- Jak a proč dokumentovat

Metodiky

- Unfied process
- Robustní x agilní přístup
- Zaměřeno na procesy nebo na lidi?
- RUP, SCRUM, TDD, EP, …

Informační systém prakticky

- Vyvíjet nový?
- Nasadit existující?
- Nasazení informačního systému je zřídkakdy izolovaná úloha.
-
- Jak zákazníka přesvědčit?
- Jak zákazníka nenaštvat?

Cvičení – návrh a implementace

- Minimalizovaný rozsah
- Složitější návrh a architektura
- Omezené využití technologií
- Minimální dokumentace
- **Revidovaná zadání v týdnu od 26. 9. 2021**

Požadavky

- Sedm artefaktů (věcí) průběžně konzultovaných na cvičení.
- Forma a obsah jednotlivých artefaktů, bude průběžně diskutována.
- Artefakt označuje *libovolný objekt nebo proces, který vznikl lidskou aktivitou, na rozdíl od předmětů přírodních* (Julian Huxley).

Artefakty I

- [2/1] Vize (dokument popisující systém z pohledu zákazníka).
- [7/4] Funkční specifikace (use case model - popis jednotlivých případů, use case diagram, diagramy aktivit).
- [3/2] Technická specifikace (první model domény, podklady pro technologická rozhodnutí, zvolené technologie a postupy).
- [2/1] Skica (wireframe, prototyp) uživatelského rozhraní.

Artefakty II

- [7/4] Návrh doménového modelu (třídy, vztahy, interakce - statický diagram tříd, sekvenční diagram, použité vzory).

- [3/2] Popis architektury systému (rozložení a propojení logických a fyzických vrstev, diagram komponent).

- [18/9] Konzistentní funkční část vybraného informačního systému s vysokým důrazem na architekturu a návrh (rozvrstvení, návrh v jednotlivých vrstvách, vzory). Předpokládá se implementace jednoduchého uživatelského rozhraní, alespoň pěti netriviálních use case a použití dvou způsobů uložení dat.

Úkoly na cvičení

- Zopakovat jednoduchý objektový návrh
- Zopakovat UML
- Zadání úkolu – diskuze