Szoftverarchitektúrák alapjai

A szoftverarchitektúra a szoftverrendszer magas szintű szerkezetére utal, amely meghatározza, hogyan kommunikálnak az egyes komponensek és modulok egymással. Az architektúra célja a rendszerek karbantarthatóságának, újrafelhasználhatóságának és méretezhetőségének biztosítása.

Rétegekre bontás (Layered Architecture)

- **Definíció**: A rétegelt architektúra a szoftver különböző funkcióit különálló rétegekbe szervezi. Minden réteg csak a közvetlen alatta lévő réteggel kommunikál, ami jól szervezett és karbantartható rendszert eredményez.

Példák a rétegekre

- **Prezentációs réteg**: Ez a réteg felelős a felhasználói interfészért és az interakciók kezeléséért. Például egy webes alkalmazásban ez lenne a front-end rész (HTML, CSS, JavaScript).
- **Üzleti logikai réteg**: Ez a réteg tartalmazza a rendszer üzleti szabályait és funkcióit. Ez a réteg dolgozza fel a felhasználói kéréseket, és itt történik az adatfeldolgozás.
- **Adathozzáférési réteg**: Ez a réteg felelős az adatok mentéséért és lekérdezéséért az adatbázisból, így elválasztja az adathozzáférést az üzleti logikától.

Előnyök: Könnyű karbantartás, különálló fejlesztés a rétegek között, jól definiált interfészek és nagyobb újrafelhasználhatóság.

Kliens-szerver modell

Definíció: A kliens-szerver modell egy olyan struktúra, ahol a kliens eszközök küldik el a kéréseiket egy központi szervernek, amely feldolgozza a kéréseket és visszaküldi az eredményt.

Fő jellemzők:

- **Kliens**: Az a végpont, amely a felhasználói kéréseket továbbítja a szerver felé, például egy böngésző.
- **Szerver**: A központi entitás, amely a kérések feldolgozását végzi, például egy webes szerver, amely a kérések alapján adatokat szolgáltat vagy feldolgozást végez.

Példa: Webes alkalmazások, ahol a böngésző (kliens) elküldi a kéréseket a szervernek, amely HTML oldalakat vagy adatokat küld vissza.

Előnyök: A központosított szerver egyszerűsíti a karbantartást és biztonsági felügyeletet, a kliens és szerver komponensek különböző helyeken futtathatók.

Többrétegű architektúrák (n-tier Architecture)

Definíció: Az n-tier architektúra kiterjeszti a kliens-szerver modellt, több rétegre bontva a rendszert,

így a kliens és szerver között további rétegek, például köztes szoftverek helyezhetők el.

Példa: 3 rétegű architektúra, ahol az alkalmazás három fő rétegre bomlik:

- **Prezentációs réteg**: A felhasználói interfészt és megjelenítést kezeli.
- **Üzleti logikai réteg**: A fő üzleti logikát és folyamatokat futtatja.
- **Adatbázis réteg**: Az adatok tárolásáért és kezeléséért felelős

Előnyök: A rétegek közötti feladatok elkülönítése növeli a skálázhatóságot és a biztonságot, mivel minden réteg csak a közvetlenül szomszédos rétegekkel kommunikál. Az egyes rétegek függetlenül fejleszthetők és tesztelhetők.

Mikroszervizek (Microservices Architecture)

Definíció: A mikroszervizek architektúra a rendszert kisebb, független szolgáltatásokra bontja, amelyek külön-külön fejleszthetők és futtathatók.

Fő jellemzők:

- Minden szolgáltatás egy adott feladatot lát el, és önállóan fejleszthető, tesztelhető és méretezhető.
- A szolgáltatások API-kon keresztül kommunikálnak egymással, tipikusan HTTP protokoll segítségével.

Példa: Egy nagyobb webes alkalmazás, ahol a felhasználókezelés, a rendeléskezelés és a számlázás külön-külön mikroszervizként működik.

Előnyök: Fokozott rugalmasság, könnyebb skálázás és az egyes komponensek független fejlesztése és karbantartása.

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:infrendalapjai_architekturak:architekturak:szoftver_architekturak

Last update: **2024/11/13 18:34**