Modellvezérelt fejlesztés

Low-code és no-code

Munkácsi Dávid
Head of DACH Region

MADIS CONSULTING
information & software services

Bemutatkozás

Munkácsi Dávid

Head of DACH Region

MADIS Consulting

Technológiai innovációs partner
Egyedi szoftveres és adatvezérelt megoldások
Nearshore és onsite üzleti modellek
Ügyfelek: hazai és DACH régió

Domének

Bank, biztosítás, ipar 4.0 (automatizálás, IoT, ipari gyártás), egészségügy, logisztika

Pozíció

MADIS Consulting németországi fióktelepének és DACH régió onsite tevékenységének vezetője

Szakmai háttér

IT Architect - .NET és web technológiák, elosztott rendszerek

Agenda

1. **Problémafelvetés és terjedelem**
   - Mire kapunk választ az előadás során?
   - Célközönség
2. **Terminológia**
   - Modellvezérelt fejlesztés
   - Low-code és no-code
   - Mi a különbség?
3. **Teljesítmény- és minőségmutatók**
   - Mikre nyújthat megoldást az MDD, low-code és no-code?
   - Mi a célfüggvényünk?
4. **Kontextus -** Esettanulmányok bemutatása
   - No-code platform fejlesztés – kutatási projekt
   - Low-code platform fejlesztés – dinamikus dashboard egy gyártási folyamat digitalizációjára
5. **Tapasztalataim**
   - Szakmai és üzleti megközelítés
   - Szoftver teljes életciklusa során előnyök és hátrányok
6. **Konklúzió és útravaló**
   - Milyen esetben ajánlott/nem ajánlott az ismeretett metódusok használata?
   - Jó tanács: modell-engedélyezési folyamat

Problémafelvetés és Scope

Problémafelvetés

”Hasonló alkalmazást már láttam...”

Újrafelhasználhatóság

Karbantarthatóság, hatékony fejlesztés

“Van erre költséghatékonyabb megoldás?”

“Szeretném, ha a szoftverem bármilyen igénynek megfelelne.”

Rugalmasság/testreszabhatóság

Célközönség

- Fejlesztők
- Szakmai vezetők
- Projektmenedzserek
- Stakeholderek
- Vállalkozók

? Modellvezérelt fejlesztés és hasonló metódusok (low-code, no-code) válasz lehet erre?

? Mire számíthattok a szoftver életciklusa során? (kihívások, előnyök, hátrányok)

Terminológia

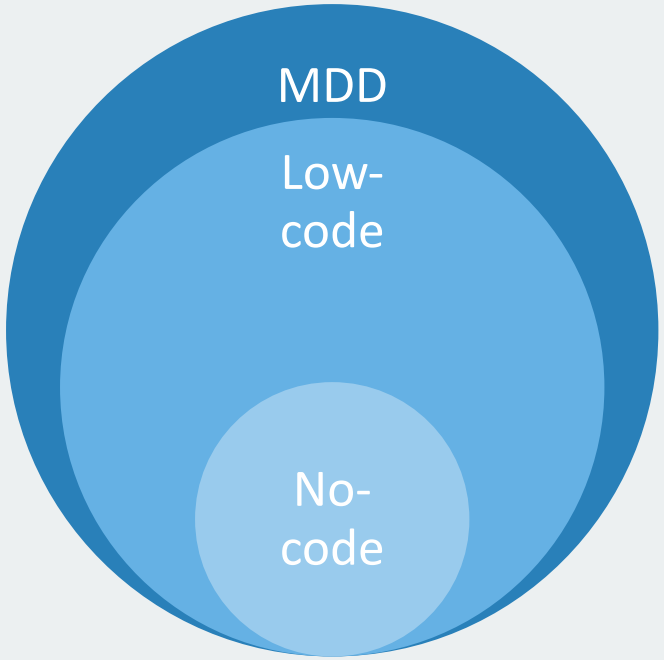
„Low-code is the latest attempt to reduce the amount of manual coding required to develop a software application. This is the same goal we have been chasing since the beginning of software engineering”

Modellvezérelt fejlesztés (MDD)
Fejlesztési paradigma
Fejlesztési folyamat központja: model
Példa: Üzleti és fejlesztői oldal között megállapodás – modell
Modell párhuzamos fejlesztése

Low-code fejlesztés
Alkalmazásfejlesztési metódus
Vizuális elemek, drag & drop, WYSIWG
Modulok integrálás kódolással/szkripteléssel

No-code fejlesztés
Kizárólag vizuális elemek
Nincs kód/szkriptelés
Alkalmazás tervezői nem írnak kódot

Konklúzióm/hipotézisem
modellvezérelt fejlesztés, no-code és low-code szinonimának tekinthetőek



Teljesítmény- és minőségmutatók

Mikre nyújthatnak megoldást az ismertetett fejlesztési metódusok?

Kontextus

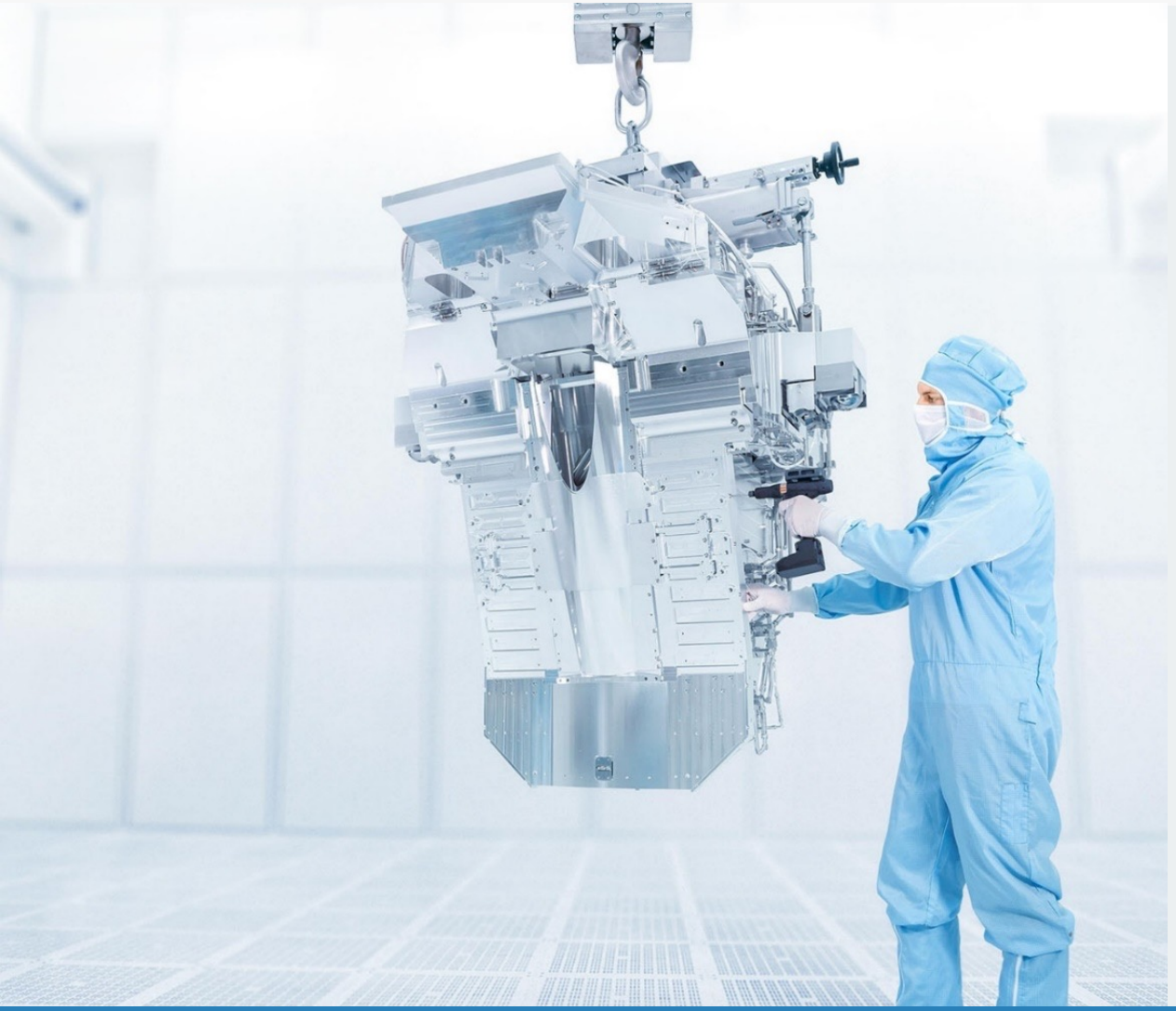
Esettanulmányok bemutatása

#1

- Kutatási projekt, német kutatóintézetnél
- Cél: **futási időben testreszabható** dinamikus dashboard
- Model: Rendszer leíró dokumentáció (API)
- Dashboard generálás; **felhasználó testreszabhatja a felületet** (méretezés, elhelyezés, ...)
- Megvalósítás: **No-code platform** tervezése és fejlesztése
- Fő karaktereresztikák: **teljes testreszabhatóság**, drag & drop, WYSIWYG

Kontextus

Esettanulmányok bemutatása



#2

- Ügyfél: piacvezető félvezető technológiákat gyártó cég
- Cél: Dinamikus dashboard egy gyártási folyamat digitalizációjára (vizualizáció és végrehajtás)
- Model: domén model - üzlet és fejlesztői csapat által tervezve
- Futási időben gyártási számítások integrálása és ennek megfelelő változások átvezetése a modellbe
- **Mérnökök saját modelljeikkel akarják irányítani az alkalmazást**
- Megvalósítás: **low-code platform** tervezése, fejlesztése, üzemeltetése
- Fő karakterisztikák: rugalmasság, újrafelhasználhatóság, **új termékek integrálása ráfordítás minimalizással**

Tapasztalataim

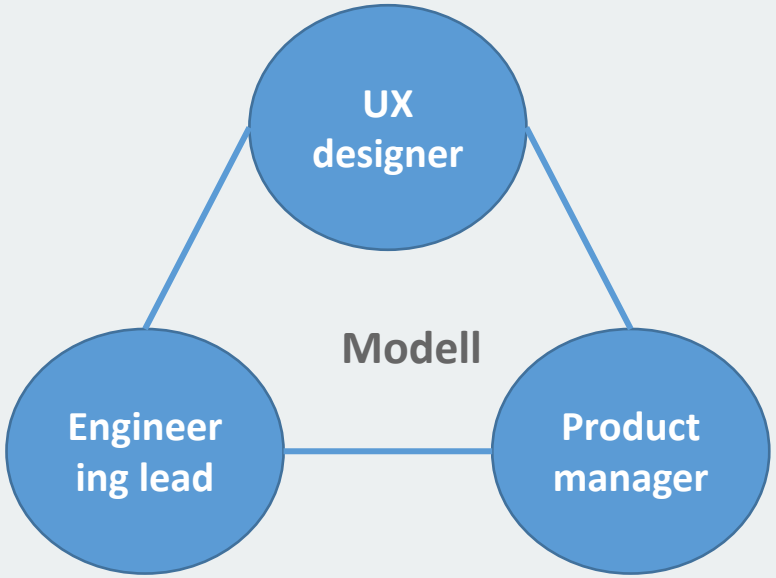
Előnyök

- Gyors feature iteráció
  - Viselkedésbeli és vizuális változtatások fejlesztői erőforrás nélkül – akár feature szinten
  - Felhasználó testreszabhatja a szoftvert
- Automatikus funkcionalitások
  - Jól lefejlesztett generikus modulok
  - Modellben implikált de nem használt funkcionalitások
  - Új feature-k a modulok kombinálásával
- Domén modellt a szakértők fejlesztik
  - Kevesebb hiba potenciál
  - Folyamatos visszajelzés
- Fejlesztői elégedettség
  - Komplex és egyedi problémák
  - Munkaerő megtartás

Tapasztalataim

Hátrányok

- Kezdeti költségek
  - Dinamikus modulok komplexek
  - QA, Design költségesebb - sok eset lefedés
  - Technológia kiválasztása nem triviális
- Nem univerzális
  - Komplex üzleti logika nehezen vagy nem lefejleszthető
- Feszültség a Product/Design/Engineering hármas között
  - Product – nélkülözhetetlen a jelenlét, nehézségek a megértésben
  - Design - Vizuális elemek testreszabása könnyen Design guideline-okba ütközhet
  - Engineering – nagy felelősség, kognitív terhelés
- Fejlesztői erőforrás nem nélkülözhetetlen
  - Modulok kombinálása
- Fejlesztői eszköztár
  - Nagy szaktudás, technológiák erős ismerete
  - Teljesen más gondolkodást igényel – architekturális, absztrakt
  - Nehéz staffing



Konklúzió

Ajánlott

- Adatvizualizációs szoftverek
- Repetitív megjelenési logikák esetén
- Magas testreszabhatóság – sok folyamatbeli vagy felületi változtatás
- Agilis környezet

Nem ajánlott

- Komplex üzleti logika/folyamatok
- Robusztus és magas hibatűrésű rendszerek
- Kevésbé reszponzív ügyfél/üzleti oldal – nem agilis környezet

Jó tanács

modell-engedélyezési folyamat (fejlesztés vagy használat)

- Modell változások aktív monitorozása
- Bizottság megszervezése: üzlet, szakmai vezető, designer

MADIS CONSULTING

information & software services

Köszönöm a figyelmet!

További információért keresse…

Munkácsi Dávid +49-174-1737532 munkacsi.david@madis.hu www.madis.hu