Fehér Péter
José M. Conejero, Pedro J. Clemente, Roberto Rodríguez-Echeverría, Juan Hernández és Fernando Sánchez-Figueroa cikke alapján

MODELL ALAPÚ MEGKÖZELÍTÉS TESZT ÚJRAFELHASZNÁLÁSHOZ INTELLIGENS OTTHON ESETÉN

Intelligens otthonok (Smart Home)

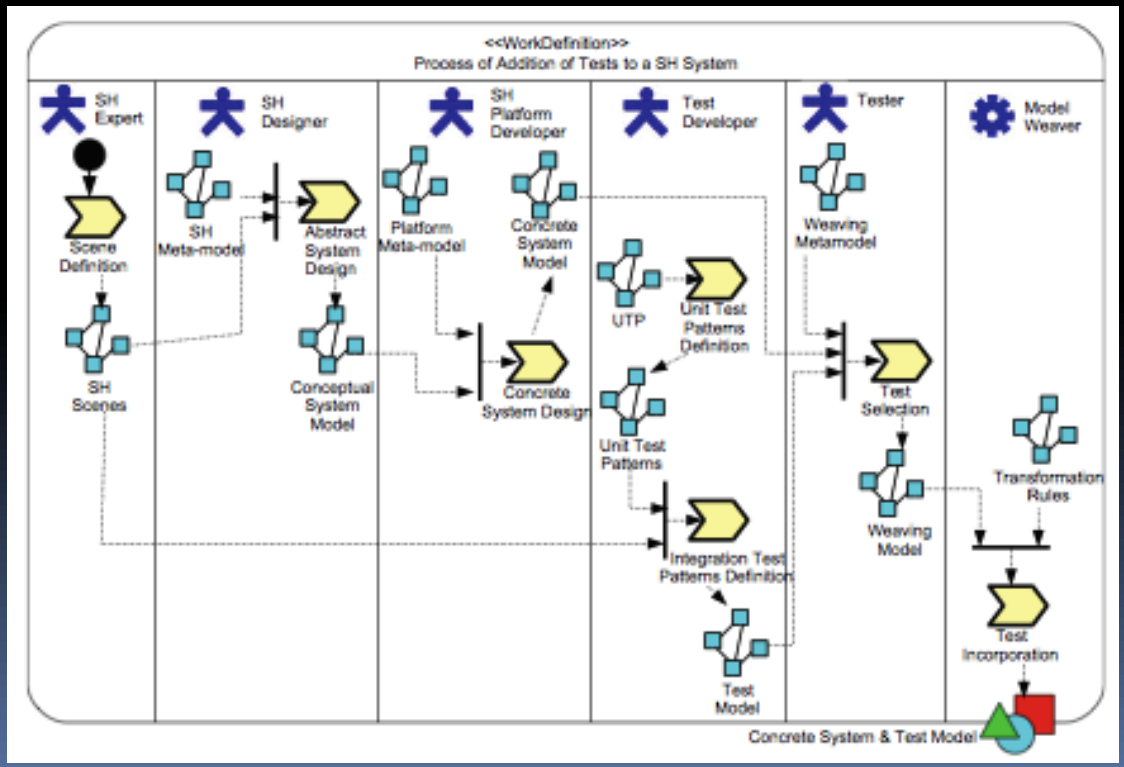
- Egyre inkább terjednek
- Magasabb absztrakciós szint hiánya
  - gyengébb minőség
  - nagy komplexitás
  - nehézkes karbantartás
  - Nem újrafelhasználható
- A modellek ugyanakkor pont hasonló problémákra kínálnak megoldást
- A Model-Driven megközelítés egyre inkább terjed a SH fejlesztések körében is
  - AMPLE project

A probléma

- Általában ezen rendszerek fejlesztése alacsonyabb absztrakciós szinten, az éppen aktuális technológiához kötötten történik
- Model-Driven Development segített ezen a problémán, növelve az újrahasználhatóságot, kezelhetőséget, csökkentve a komplexitást
- Az MDD azonban nem fedi le a teljes fejlesztési ciklust, a tesztek még mindig az utolsó fázisban történnek -> rendszerint nem újrafelhasználhatóak

Három fő fázis

- Intelligens otthon rendszermodellezés
- Teszt pattern modellezése
- Teszt kiválasztása és beépítése a rendszerbe
- Az első két fázis egymástól teljesen függetlenül végezhető

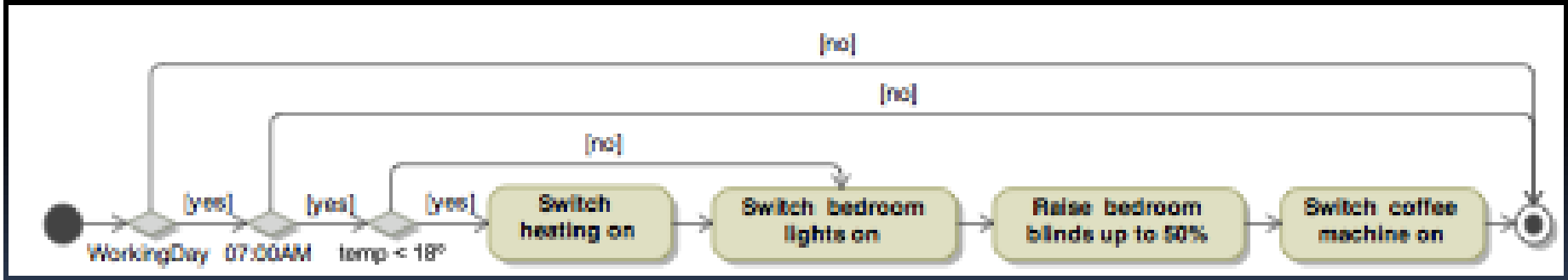


I. fázis – Intelligens otthon modellezés

- Esemény (scene) pontos, részletes definiálása
  - Az egész rendszert több „scene” együttese alkotja
- Absztrakt rendszer kialakítása
  - Abstract Smart Home Meta-Model (SHMM) -> koncepcionális rendszermodell
  - Nagyon magas absztrakciós szint -> platformfüggetlenség
- Konkrét rendszer kialakítása
  - Az SHMM kibővítése konkrét platformra (pl. KNX), ezáltal elérhető pl. az automatikus kódgenerálás

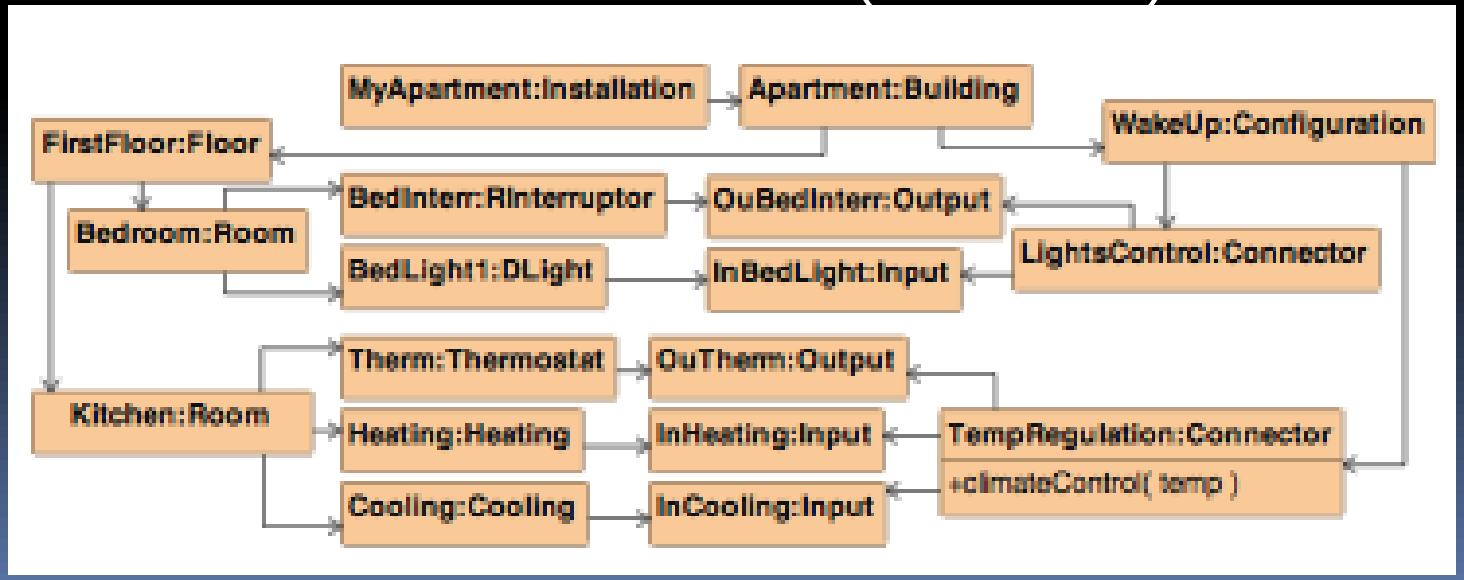
Scene definiálás – WakeUp scene

- UML activity diagram segítségével
- A megfelelő feltételek esetén négy tevékenység automatikus megvalósítása



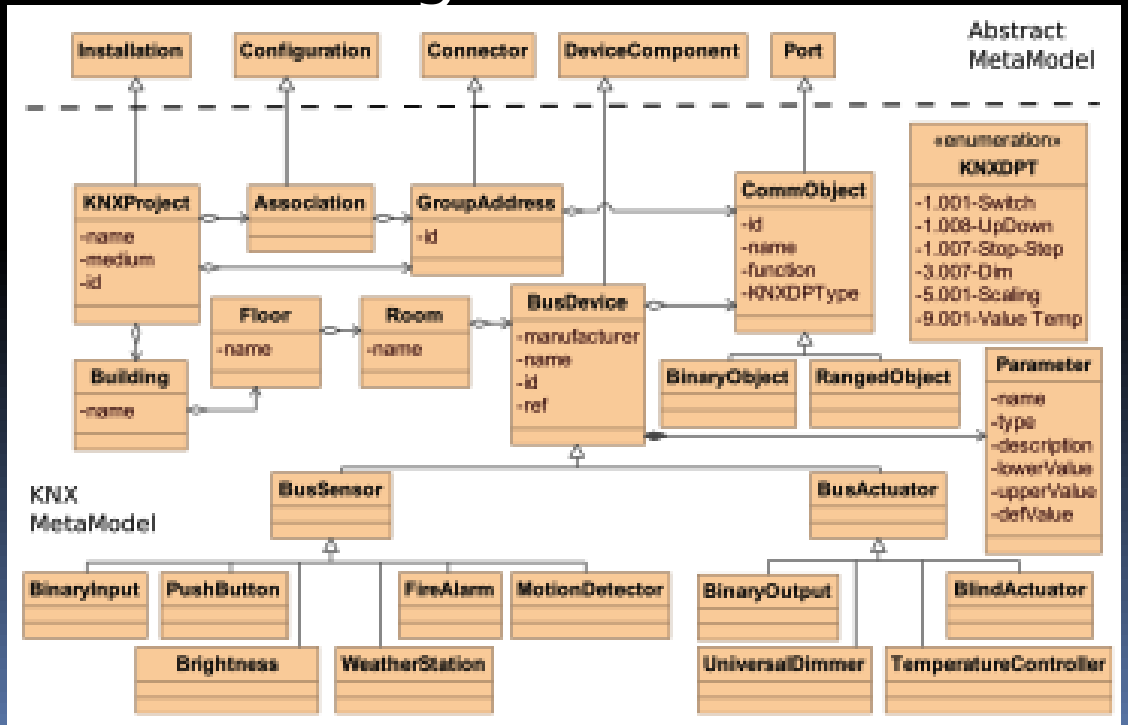
Absztrakt rendszermodell kialakítása

- Az SHMM az ECORE meta-modell segítségével lett kialakítva, ami része az AMPLE-nek
- Minden információt tartalmaz (installáció, komponensek, helyszínek)
- A *Connector* osztály teszi lehetővé a kommunikációt a szenzorok és a szerkezetek (actuators) között



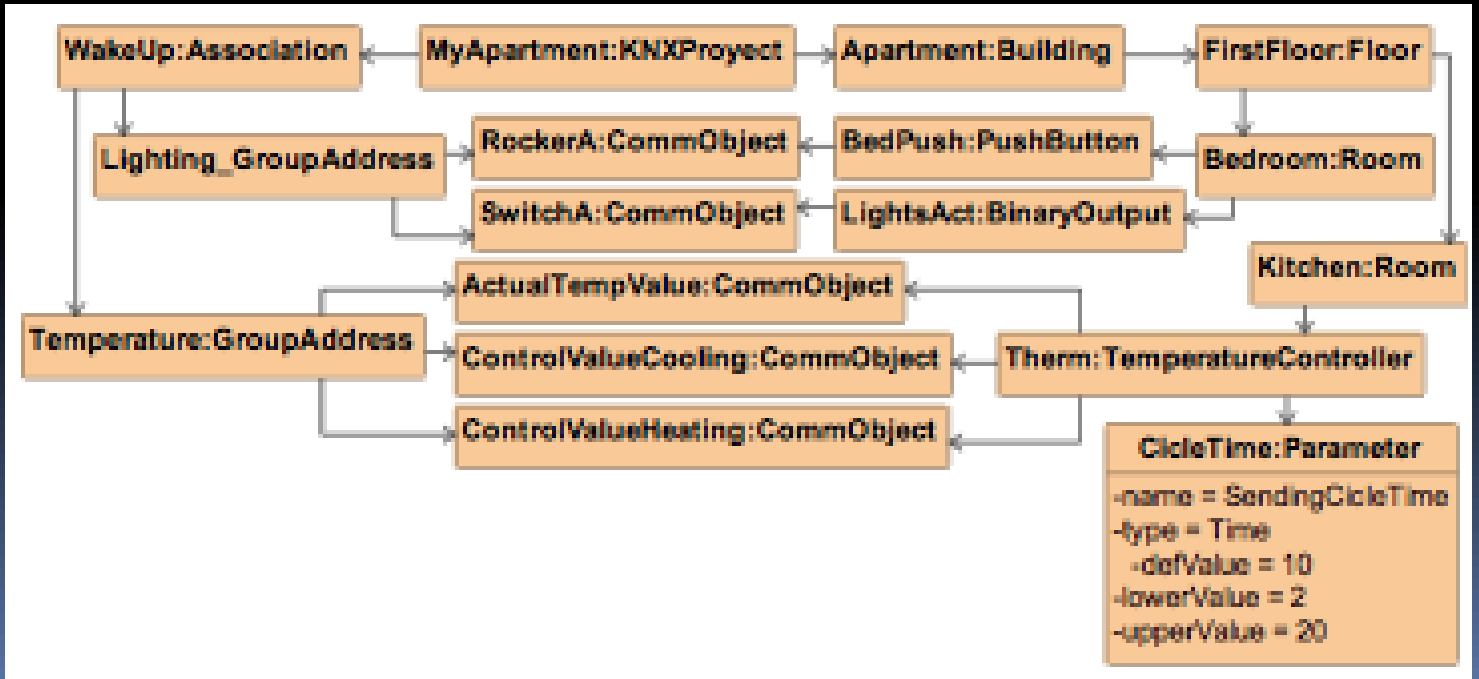
Konkrét rendszer kialakítása

- Az SHMM-ben lévő entitások kiegészítése új, KNX metamodellbeli (KMM) entitásokkal
- A specifikus metamodell definiálása az absztrakt kiegészítéseként -> a teszt patternek is megvalósíthatóak lesznek



Konkrét rendszer kialakítása

- KNXProject -> root class
- Association class -> modeled scenes
- Group Address class -> érzékelő-szerkezet kapcsolat

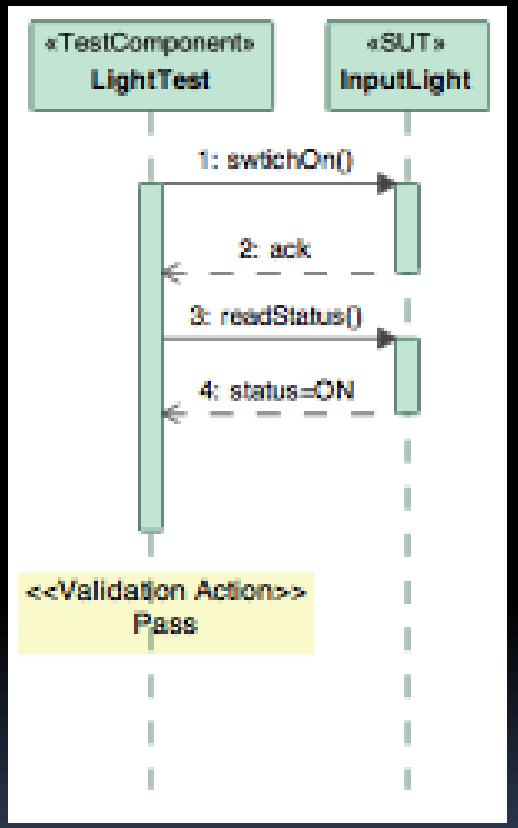


II. fázis – Teszt patternek kialakítása

- A teszt esetek patternként való modellezése lehetővé teszi, hogy más SH rendszerek esetén is felhasználhatók legyenek
- Ezen patternek sztenderd Testing Meta-modellen (TMM), konkrétan az UTP-n alapszanak
- Két különböző teszttípus
  - Unit Tests (-> individual device testing)
  - Integration Tests (->scene behavior testing)
- A patterneket egy repositoryban tároljuk

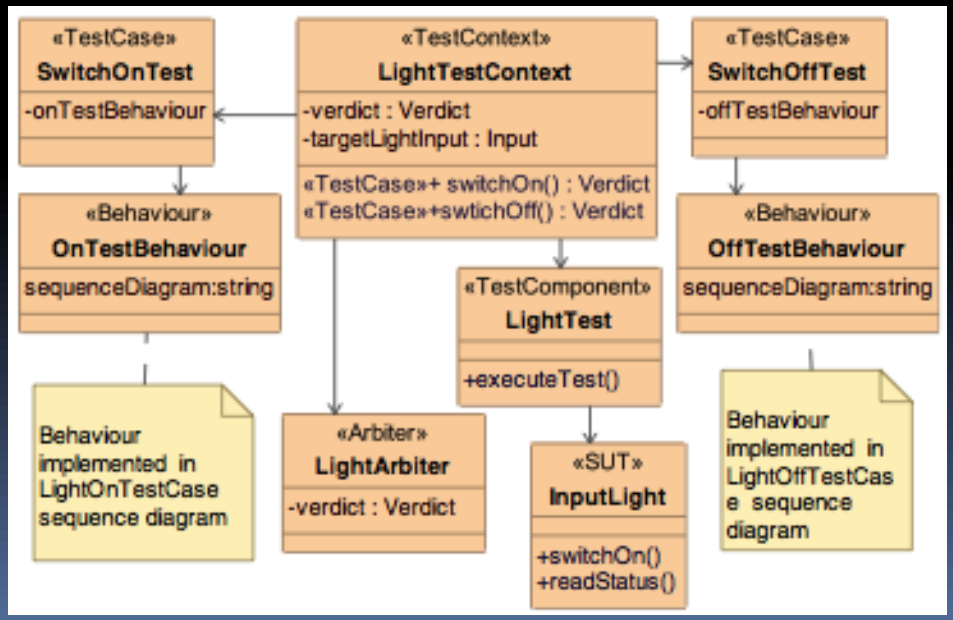
Unit Testek definiálása

- Szekvencia diagram, ahol a működést modellezzük
- Class diagram, ahol a role-okat határozzuk meg
- *Validation Action* -> kiértékeli a teszt eredményét, s meghatározza az „ítéletet”



Unit Testek definiálása – Test Context

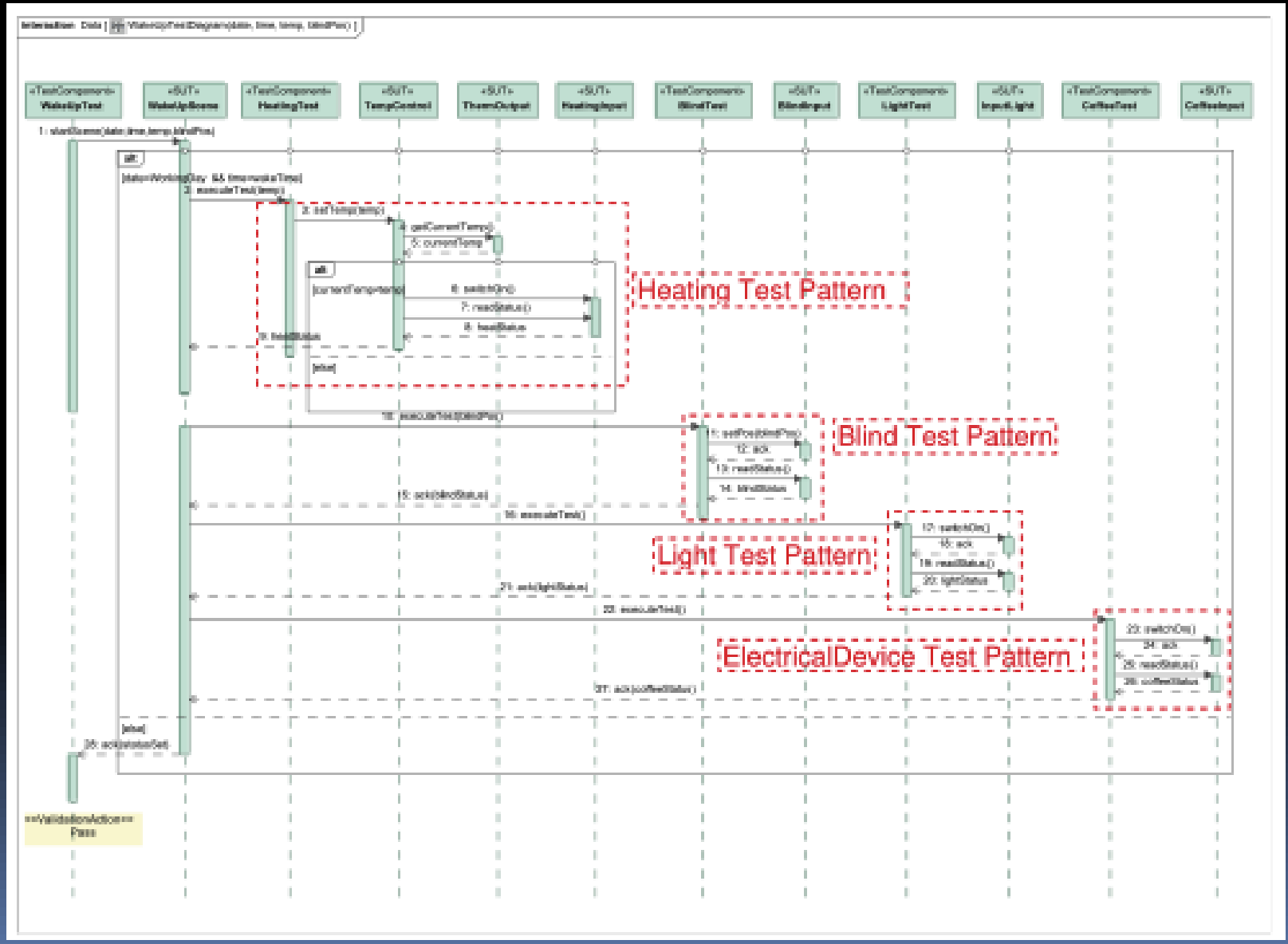
- Több teszteset, mindegyikhez külön szekvencia diagram
- Egy teszt komponens, mely a futtatásért felel
- A SUT nem kapcsolódik semmilyen specifikus objektummal sem!



Integration tesztek kialakítása

- Az eszközök közötti kapcsolatokat teszteli, hogy az eszközök megfelelően reagálnak a jelekre
- Szekvencia diagram használata
- A Unit Testek felhasználhatóak
- Paraméterek megadása (day, time, predefined temp, blind position)
- Pl.: (Monday, 07:00AM, 19, 50)

Integration test example



III. fázis – Tesztek kiválasztása, beépítése

- Tesz kiválasztása
  - Ún. Weawing modell kialakítása a SH rendszerhez a Weawing Metamodel (WMM) segítségével
- Ennek segítségével a tesztek automatikusan kialakíthatóak a konkrét rendszerben

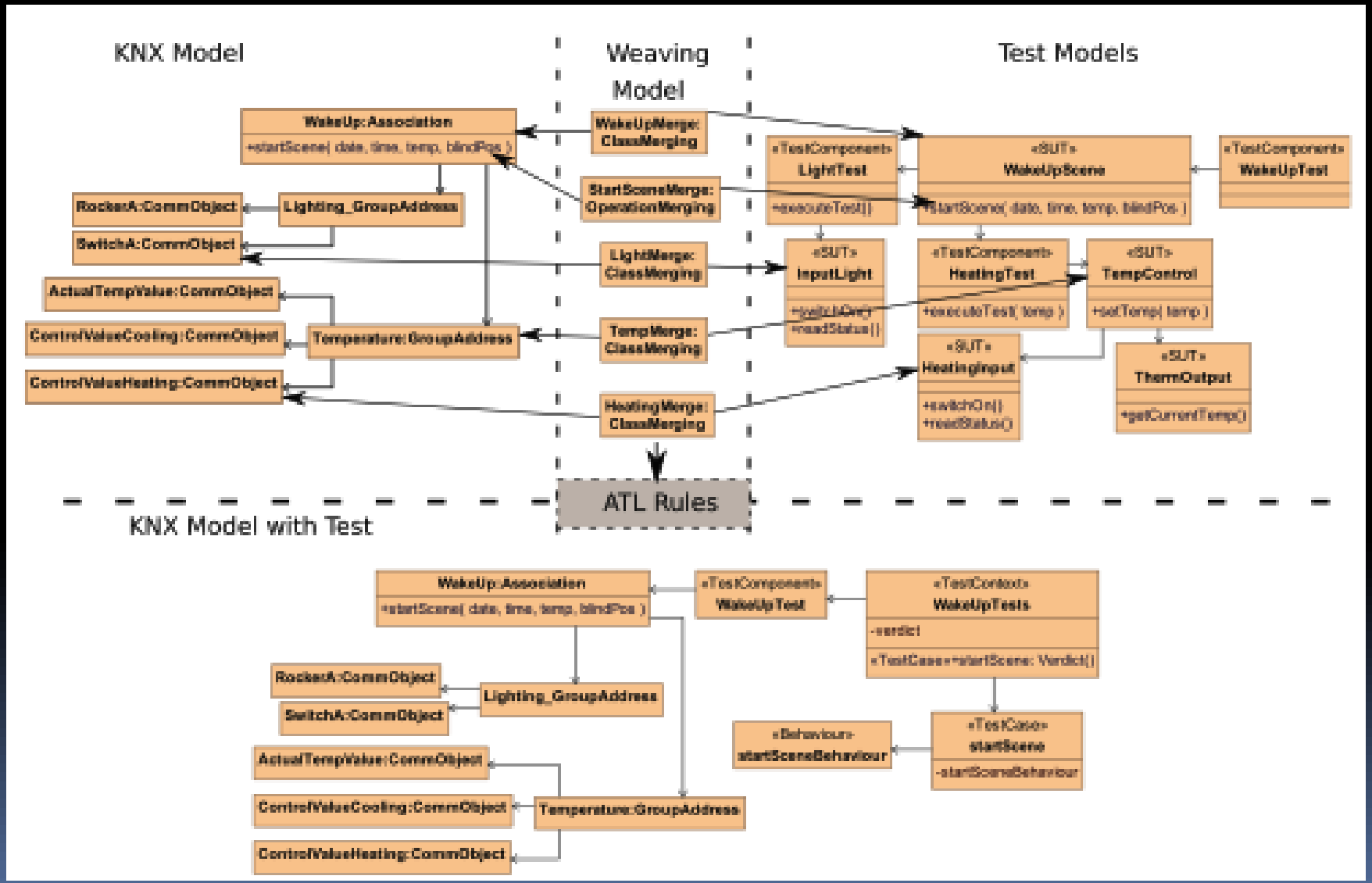
Az összefűzés

- A test patternek kiválasztása a korábban definiált funkcionalitásokon és a scene-eken alapszik.
- Amint a teszteket kiválasztásra kerültek, automatikusan összefűződik az SH model és a Test Model
- A kiválasztás és az összefűzés érdekében:
  - Weaving Metamodel a SH domainhez, mint AMW metamodell kiegészítés
  - Összeillesztő szabálygyűjtemény kidolgozása

Az összefűzés

- Weaving Metamodel
  - Előkészíti a környezetet
  - Egymáshoz illeszti a SH modell és a Test modell megfelelő elemeit
  - Meghatározza a típusokat a SH modellben, melyek részt vehetnek a Unit és Intergation tesztekben
  - Kialakítása a SHMM alapján -> újrafelhasználható
- ATL szabályok kialakotása
  - Merging Classes (Rule1)
  - Merging Operations (Rule2)
  - Not-Merging Classes (Rule3)

Az összefűzés



Konklúzió

- Folyamat, mely automatizálja a tesztek generálását a SH rendszerekbe
- A folyamat a már egyre terjedő MDD része
- A tesztek így újrafelhasználhatóak, hisz immár modell szinten vannak, platformfüggetlenek
- A tesztek patternként vannak definiálva, ezzel is segítve a többszöri felhasználást
- Hatékony összefűzése a teszteknek és konkrét rendszereknek