# Modell alapú hibainjektáló környezet fejlesztése

## Önálló laboratórium feladat összefoglalója (2. félév)

**Oláh János (FAXPFU)**
**Konzulens: Majzik István**
**BME Méréstechnika és Információs Rendszerek Tanszék**
**Informatikai infrastruktúra tervezése szakirány, 2007/2008. II. félév**

Számítógépes szoftverek készítésének, illetve elkészült programok összevetésének fontos szempontja a szolgáltatásbiztonság mérése. A program robusztusságának, az alkalmazott hibakezelési technikák hatékonyságának értékelését nagyban megkönnyítik az olyan hibainjektáló keretrendszerek, melyek segítségével a program tetszőleges helyére injektálható hiba, meghatározott terhelés (workload) mellett, valamint a monitorozás is megfelelően támogatott. Java nyelvű környezetben ezt a feladatot hatékonyan támogatja a Javassist nevű osztálykönyvtár, mely a Java class fájlokhoz biztosít bytcode szintű hozzáférést.

Az e félévi önálló laboratórium feladatom megkezdéseként egy, a hibainjektáló környezetek megvalósításához ajánlott mintakészlettel (pattern system) ismerkedtem meg, mely segítséget nyújtott a későbbi önálló munkában is.

A Javassist lehetőségeinek feltérképezése után elkészült egy terv az általam készítendő hibainjektáló keretrendszer fő funkcióiról és korlátairól. A fő funkciók meghatározásakor a korábbi félévben megismert benchmark rendszerek is például szolgáltak. A tervezett funkciók nagy vonalakban a következők: A felhasználó egy grafikus felületen láthatja a vizsgálandó rendszer általa kiválasztott tartományát, és ezen a felületen keresztül egy tetszőleges hibakonfigurációt tud összeállítani. A hibákkal kiegészített (instrumentált) rendszert ezután a keretrendszer képes futtatni, és a hibák hatását megfigyelni, illetve a futás közben begyűjtött információkat adatbázisba kimenteni a későbbi elemzések kedvéért. Fontos szempont a hibakonfigurációk eltárolása a kísérlet későbbi megismételhetőségéért, illetve a releváns adatok tárolása, hogy (például a fejlesztés alatt álló) programok különböző verzióit, vagy azonos feladatra alkalmas szoftverek eredményeit könnyen össze lehessen hasonlítani.

A keretrendszer funkcióinak specifikálása után a keretrendszer elkészítéséhez használható technológiák keresésével és kipróbálásával foglalkoztam. A grafikus megjelenítéshez használható technológiák kipróbálása megtörtént, de itt végleges döntés még nem született, több konkurens technológia is megfelelőnek látszik.

Ezután következhetett a keretrendszer architektúrájának megtervezése. A tervezéshez felhasználtam a már korábban is említett mintakészletet, szükség volt Javassist-specifikus tervezési megfontolásokra, illetve a felhasználandó technológiák lehetőségeinek figyelembe vételére (különös tekintettel az Eclipse RCP által nyújtott szolgáltatásokra).

A szoftver tervezési lépések után az architektúrában definiált egyes komponensek elkészítését kezdtem el. A munka során eddig két fő komponens (parser component, fault-injector component) készült el. Így az eddig meglévő komponensekkel lehetőség van a megadott java class fájlok feldolgozására, és megjelenítésére XML fájlként, illetve egyelőre egy kézzel készített hibaleíró fájl segítségével hibák injektálására is.

A munka folytatásként a hibák hatását monitorozó komponens elkészítése következik, majd így a fő komponensek elkészítése után ezek integrálása egy rendszerbe, mely tehát Eclispe RCP alkalmazásként fog elkészülni.