Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Tesztoptimalizáló eszköz fejlesztése

Molnár Gábor I. évf, (MSc) villamosmérnök/mérnökinformatikus szakos hallgató
Konzulens: dr. Micskei Zoltán Imre okl. mérnök-informatikus, PhD
Adjunktus, MIT
Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 2. összefoglaló
2014/15. I. félév

Regressziós tesztelés során a korábban elkészített tesztkészletet futtatjuk, a megváltozatott programverzión, ezáltal meggyőződhetünk arról, hogy a rendszer korábbi funkciói továbbra is működnek. Azonban egy ilyen tesztkészlet futtatása nagyobb rendszerek esetén időigényes.

A tesztoptimalizálás ezzel a problémával foglalkozik, nevezetesen milyen módszerekkel lehet ezt a futtatási időt csökkenteni. Ilyen módszer többek között a tesztkiválasztás, amikor a módosítások ismeretében meghatározzuk a szükségesen futtatandó teszteket.

A szakirodalomban többféle algoritmus is rendelkezésre áll, azonban jelenleg nem ismert egy olyan eszköz, ami akár a legalapvetőbb optimalizációs feladatokat képes lenne ellátni.

Az önálló laboratóriumom témája tehát egy ilyen eszköz fejlesztése volt.

Az optimalizációs problémát három komponensre bontottam:

1. Kell komponens, ami az aktuális programozási platformnak megfelelően képes általános teszt-komponens modellt építeni (pl. Java kódból és az ismert módosításokból).
2. Egy másik komponens az általánosított modellen elvégzi a tesztoptimalizálást.
3. A harmadik komponens pedig képes végrehajtani az optimalizált tesztkészletet (pl. meghívja a JUnit tesztfuttatót).

Ebben a félévben a második, optimalizáló komponens készült el. Ez a komponens Eclipse Modelling Framework (EMF), valamint Eclipse plugin alapú. A felépített teszt modelleken EMF-IncQuery-vel azonosítja a szükségesen vizsgálandó komponenseket és teszteket, majd ezeken egy halmazfedési algoritmussal fedést keres.

Mivel a minimális fedő halmaz keresése NP-teljes probléma, így egy mohó algoritmus alapú heurisztikát alkalmaztam, ahol a költségfüggvény a tesztek futásideje.

Összességében a félév során elkészült egy átfogó architektúra-terv, illetve megvalósítottam ennek a központi részét. Az elkészült eszköz képes komponenseket fedő teszthalmazokat keresni, illetve felismerni azt, hogy ha egy komponens nem lefedhető, azaz további tesztek írása szükséges.

További munkát tesz lehetővé a különféle adapter-komponensek implementációja, például kiterjeszthető az eszköz olyan módon, hogy Java forráskód alapon épüljenek a teszt–komponens fedési modellek, illetve az optimalizált tesztkészlet futtatható legyen JUnit-tal. Ezen felül megvalósítható még az egész toolchain Eclipse illetve valamilyen build rendszerrel történő integrációja is, így például Maven-nel vagy akár közvetlenül Jenkins-szel is.