Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Automatikus szoftvertesztelési eszközök .NET-hez

Hugyi Károly (HJ3IUV), IV. évf, (BSc) mérnök inf. szakos hallgató
Konzulens: Micskei Zoltán tanársegéd, MIT
Informatikai technológiák szakirány Rendszertervezés ágazat
Önálló laboratórium összefoglaló
2010/11. II. félév

A szoftverfejlesztés során mind a tervezésben, mind az implementáció során előfordulhatnak hibák. A tesztelés többek között ezeket hivatott kiszűrni.

A tesztelést – akár csak a szoftvert – meg kell tervezni. Ahhoz, hogy minél pontosabb tesztek készüljenek, a fejlesztőknek és tesztelőknek már a projekt elején együtt kell dolgozniuk. Tesztelés során is fontos a megfelelő dokumentáció készítése.

Különböző tesztelési szintek vannak:

- Komponens tesztelés: a szoftvert alkotó modulok működésének tesztelése
- Komponensek integrációjának tesztelése: az egymással kommunikáló modulok helyes működésének tesztelése
- Rendszerteszt: a teljes szoftver (közel) végleges állapotában történő tesztelés. Itt vizsgáljuk azt is, hogy más rendszerekkel, pl. az operációs rendszerrel helyesen működik-e együtt
- Elfogadhatóság teszt: ezt többnyire a végfelhasználók végzik. Itt már nem a hibakeresés a cél

Alapelv, hogy egy-egy hiba kijavítása után ismét le kell futtatni minden tesztet, ami kapcsolatban állt a hibát okozó komponenssel, hogy meggyőződjünk arról, hogy az javítva lett, valamint hogy nem keletkeztek újabb hibák.

Amennyiben a szoftver modelljén változtatni kell, legtöbbször az érintett részt tesztelő kódon is változtatni kell. A modell alapú tesztelésben rejlő automatizálási lehetőségekkel élve a tesztkódot néhány kattintással meg tudjuk változtatni, elég a modellbe átvezetni a változásokat, ebből a keretrendszer legenerálja az új tesztkódot.

A félév során először a tesztelés alapjaival ismerkedtem meg: tesztelési szintekkel, tesztelési módokkal. Az alapok után a Microsoft által fejlesztett SpecExplorerrel foglalkoztam. A program modell alapú tesztelési lehetőséggel egészíti ki a Visual Studiot. A C# nyelvet már korábbról ismertem, így a modellezéshez csupán néhány új osztályt kellett megismernem. Ezzel szemben a függvényhívások módját, a paramétereket, a különböző teszteseteket Cord-scriptben kell megadni, így az ezzel történő ismerkedés hosszabb ideig tartott, köszönhetően a Cord számos operátorának.

A félév második felében különböző példaprogramok segítségével mélyedtem el a SpecExplorer képességeiben, végül két általam írt egyszerű programmal igyekeztem begyakorolni a megszerzett tudást.

Figyelembe véve a Cord lehetőségeit, és a modellezés viszonylagos egyszerűségét, véleményem szerint érdemesebb a szoftverfejlesztést modell alapú tesztgenerálást használni, a hagyományos, kézzel történő helyett.