Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Automatizált egységtesztelés ipari környezetben

Honfi Dávid (B6PKQY, I. évf, MSc) mérnök informatikus szakos hallgató

Konzulensek
Dr. Micskei Zoltán adjunktus, MIT,
Vörös András tudományos segédmunkatárs, MIT,
Theisz Zoltán, Evopro Innovation Kft.

Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 1 összefoglaló
2013/14. I. félév

Eddigi munkám során megismerkedtem az alapvető tesztelési fogalmakkal és metodikákkal, illetve a Microsoft Research által fejlesztett Pex eszközzel. A Pex egy struktúra alapú tesztgeneráló eszköz a .NET keretrendszerben C# nyelven íródott szoftverek számára. Az eszköz parametrizált tesztmetódusok számára állít elő bemeneteket. A Pex képességeinek javát a dinamikus szimbolikus végrehajtás adja, amely segítségével képes felfedezni az olyan tesztelendő lefutási ágakat is, amelyek a manuális tesztelés során kimaradhatnak. Az eszközt először egy egyszerű szoftveren, majd szakdolgozatom keretében már a PetriDotNet2 modellverifikációs eszközön próbáltam ki. A bíztató eredmények hatására úgy véltem, hogy érdemes a Pex képességeit ipari környezetben is felmérni.

A félév során tehát a feladatom volt, hogy egy az Evopro Innovation Kft.-nél fejlesztett tartalomkezelő szoftver szerveroldali komponensén kipróbáljam a Pex-et és elemezzem a kapott eredményeket. Ezt követően pedig tovább folytatva a tesztelést visszajelzést nyújtsak a programfejlesztők számára a kód minőségével kapcsolatban. Az említett szoftver már egy befejezett fejlesztés, így a dokumentációk és a kód alapján kellett megismerkednem a szoftver struktúrájával. A szoftverhez a fejlesztők főleg integrációs teszteket készítettek, így érdekes volt megvizsgálni, hogy milyen hibák fedezhetők fel kizárólag egységtesztelés segítségével.

A Pex az első futtatása során nagyon kevés osztályhoz volt képes tesztesetet generálni. Az eredmények vizsgálata alapján megállapítottam, hogy a kevés teszteset legfőbb oka a megfelelő izoláció hiánya volt, azaz a külső függőségekből eredő hibák befolyásoltak a Pex működését, amely így elakadt a felfedezés egy, az egységen kívüli kódrészletnél. A kapott eredmények alapján, illetve a fejlesztőktől kapott visszajelzésekből megállapítható, hogy a Pex által megtalált hibák valódiságának aránya meglehetősen alacsony. Ez annak volt köszönhető, hogy a Pex elakadt az egyes hívási láncok elemzése során. A problémák megoldására a Pex számára Factory metódusokat hoztam létre. Végül pedig a Pex teljes működésének biztosításához az izoláció megvalósításának tervét készítettem el. Az elkészített terv alapján a szoftver egyetlen, adatbázis manipulációért felelős osztályát kell izolálnom paraméterezhetően, hogy a mock a Pex által generált bemenetek szerint működjön. Ennek a környezetnek az implementációja jelenleg is folyamatban van. Emellett párhuzamosan tervezés alatt áll a rendszer kisebb egységek szerinti izolációja, amelyet félig átlátszó izoláció segítségével tervezek megoldani, amelynek lényege, hogy a tesztelendő egység szemszögéből valójában izolált környezet biztosítható, azonban a Pex a kód további részét is felfedezi, így képes relevánsabb bemeneteket előállítani.

Konklúzióként elmondható, hogy egy ilyen méretű szoftver utólagos tesztelése meglehetősen lassú és nehézkes folyamat, hiszen meg kell ismerni a szoftver részletes felépítését a dokumentációk és a kód alapján. Emellett a nagyméretű kódbázis általi függőségek miatt az izoláció megvalósítása sem triviális kérdés.

Továbbfejlesztési lehetőségként természetesen elsőként említhető a tartalomkezelő szoftver Pex-szel való tesztelésének megvalósítása a teljes izolációs környezet felépítésével. További érdekes vizsgálatot jelenthet a fenti, felmerülő problémák alapján, hogy hogyan lehet a Pex-szel való tesztelést beépíteni az ipari fejlesztés folyamatokba inkrementális jelleggel, illetve szintén új eredmények kaphatóak az izolációs környezetek modell alapján történő generálásának vizsgálatával is.