Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

ANSI C programmodulok közötti hívások felderítése teszt fedettségi analízishez

Fejes Endre G2MMW3 1. évfolyam mérnökinformatikus MSc
Konzulens: Piroska László, Robert Bosch Kft.
Belső Konzulens: Majzik István, MIT
Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 2 összefoglaló
2010/11. I. félév

Az Önálló laboratórium 2 keretein belül a korábban Önálló labor 1 illetve szakmai gyakorlat alatt elkezdett feladatot folytattam.

Biztonságkritikus rendszereknél kiemelt fontosságúak és a fejlesztési időigényét nagyban befolyásolják a regressziós tesztek. Ezek időigényessége csökkenthető annak meghatározásával, hogy egy adott helyen történő módosítás az alkalmazás mely további részeit érinti. A nem érintett tesztek kijelöléséhez fontos információ, hogy adott tesztek a kód mely részét fedik le.

A célkitűzés egy olyan eszköz tervezése és fejlesztése volt, ami az egyes tesztek kódlefedettségének meghatározásához nyújt segítséget. A feladat konkretizálása során a vizsgálandó kódfedettséget leszűkítettük a programmodulok közötti hívások fedettségére. Ennek megfelelően a támogató eszköz elsődleges célja a modulok közötti hívások felderítése volt.

A Bosch Kft. ennek az eszköznek a használatával az automata sebességváltók mikrokontrollerén futó beágyazott, ANSI C nyelvű programok teszteléséhez szükséges idő lerövidítését tűzte ki célul.

A lefedettség meghatározásának támogatását az eszköz a program modulok interfészének illetve a modulok közötti, interfészeken keresztül történő hivatkozásoknak a felderítésével végzi. Mivel az ANSI C nem definiál modult illetve interfészt, így modulnak egy modulonkénti szöveges konfigurációs fájlban megadott fájl listát értünk. A modul interfészének pedig a modulhoz tartozó fejléc fájlokban definiált és *más modulok által hivatkozott* függvényeket illetve változókat. Ez utóbbi meghatározásához szükség van a program elemzésére.

A feladat során *Eclipse* alapú technológiákkal dolgoztam. A *Plug-in Development Environment* (PDE) nyújtott lehetőséget az eszköz integrált fejlesztő környezetbe ágyazására. A forrás állományok objektumorientált kezeléséhez a *C Development Tools* (CDT) nevű eszköztárat használtam. A nyelv elemeit *Abstract Syntax Tree* (AST) technológiával lehet leírni. Ez a nyelv elemeit tartalmazó modell. Ennek egy konkrét példánya a *Concrete Syntax Tree* (CST), ami egy adott program nyelvi alkotóelemeiből épül fel. A CDT segítségével egy, az adott C program alapján egy *Visitor* mintával bejárható CST kapható meg.

A felépült CST alapján az eszköz meghatározza a definiált elemeket, a hivatkozásokat, majd kiszűri a modulon belüli kapcsolatokat, illetve a nem használt változókat, függvényeket. Ezek alapján jelentést készít. Modulonkénti és fájlonkénti bontásban listázza az azonosított interfész elemeket. Először a modulhoz tartozó fejléc fájlokban definiáltakat, jelölve, hogy melyik másik modulban, és azon belül fájlban történt az adott elemre hivatkozás. Ezután a modulhoz tartozó fejléc és forrás fájlokban talált hivatkozásokat más modul interfész elemeire.

Ezek alapján meghatározható, hogy egy adott függvény módosítása esetén szükség van-e a hozzá tartozó modulon kívül más modulokat tesztelni.

A Önálló Labor során sikerült az eszköz megtervezése és olyan szintű implementációja, ami meghatározza, és egy jelentésben összesíti egy előfeldolgozott ANSI C moduláris program interfészeit. Szintén elkészült az eszköz végleges folyamatba való integrálásának terve, aminek megvalósításával már most hasznos információkat lehet kigyűjteni egy adott alkalmazásról.

Az Önálló Labor második féléve során sikerült az eszköz statikus ellenőrző részének véglegesítése illetve a dinamikus ellenőrzés rész megtervezése.