Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Hierarchikus állapottérképek formalizálása és ellenőrzése

Kálmán Tibor I. évf, MSc mérnökinformatikus szakos hallgató
Konzulens: Hajdu Ákos doktorandusz, MIT
Kritikus rendszerek főspecializáció
Önálló laboratórium 1. összefoglaló
2016/17. II. félév

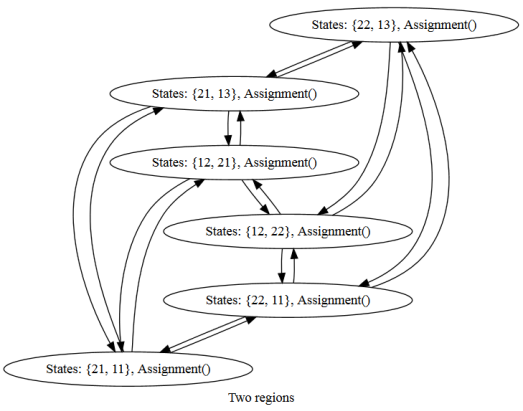
Az önálló laboratóriumi feladatom során olyan Java osztályokat kellett létrehoznom, melyekkel hierarchikus állapottérképek reprezentálhatóak. Ezen felül ezen osztályokat felhasználva explicit modellellenőrzést is készítettem, mely bejárja az állapotteret és ezt gráf formában képes számunkra visszaadni.

A feladat egy nagyobb projekt része, melynek célja az, hogy létrejöjjön egy olyan eszköztár, mely segítségével a felhasználó könnyen és magas szinten specifikálni tudja az általa kitalált megoldást állapottérképpel, melyet utána ellenőrizhet és akár kódot is generálhat belőle. Az általam készített rész ebből tehát az ellenőrzés, és az ellenőrzés (és akár a kódgenerálás) mögött meghúzódó Java modell elkészítése volt. A félév során a Modell alapú rendszertervezés tárgy kapcsán Beöthy Bencével ugyanennek az eszköztárnak egyéb részein is dolgoztunk. Én készítettem egy Sirius alapú editort, mellyel egyszintes állapottérképeket lehet grafikus felületen specifikálni, ő pedig készített egy kódgenerátort az általam írt Java modellből kiindulván és M2M transzformációt a Sirius-szal elkészített modell és a Java modell közé. A Siriusos editor a már mások által elkészített Theta Statechart EMF metamodellel dolgozik. Ezen felül készítettem egy interaktív szimulátort is, mellyel lépésről lépésre járhatjuk be a modellt, figyelhetjük az aktuális állapotkonfigurációt és tüzelhetünk tranzíciókat.

Az első lépés az irodalomkutatás volt. Ennek során feldolgoztam a Formális módszerek című tárgy releváns anyagrészeit, megcsináltam a Kritikus architektúrák laboratórium két, program verifikációval foglalkozó alkalmának feladatait és részben elolvastam Yael Meller 2016-os *Model Checking Techniques for Behavioral UML Models* című PhD dolgozatát is.

Ezután következett a Java modell elkészítése. Ennek során fő szempont volt az, hogy jól navigálható legyen és a lehető legtöbb helyen használja a Theta keretrendszer által nyújtott lehetőségeket (pl. kifejezések reprezentációja, értékadások kiértékelése). Természetesen ehhez arra is szükség volt, hogy a keretrendszerrel részben megismerkedjek. Implementáltam az állapotba be- és kilépéskor értékadást vagy eseményt (és ezek kombinációjának) kiváltását, guard feltételt és akciókat a tranzíciókon,

Végül magát az explicit modellellenőrzést készítettem el. Ennek során a fő kihívás az állapotkonfigurációt leíró osztály és a tranzíciók tüzelését megvalósító függvény elkészítése volt. Utóbbi a vártnál lényegesen bonyolultabb logikával rendelkezik, a tetszőleges mélységű hierarchia és párhuzamos régiók megléte miatt. Működése során szélességi bejáráshoz hasonlóan térképezi fel az állapotteret és Graphviz alapon képes ábrázolni azt.



Two regions

Ez az ábra egy példa arra, hogy milyen kimenetet készít az általam írt algoritmus. Itt egy igen egyszerű, kétrégiós állapottérkép állapottere figyelhető meg. Az egyik régióban 3, a másikban 2 állapot van, régión belül teljes gráfként összekötve tranzíciókkal. Jól látható, hogy minden konfigurációból (és konfigurációba) három nyíl vezet ki (és be), hiszen az első régióban mindig kettő, a második régióban pedig egy tranzíció tüzelésre kész. Az egyszerű szemléltetés kedvéért nincs guard és akció sem, se állapoton, se tranzíción.