FIATAL MŰSZAKIAK TUDOMÁNYOS ÜLÉSSZAKA

Kolozsvár, 2003. március 21-22.

NYÍLT, SVG ALAPÚ KERETRENDSZER METAMODELL- BÁZISÚ MODELLEK AUTOMATIKUS MEGJELENÍTÉSÉRE

Domokos Péter, Varró Dániel

In the paper, we propose an automated, SVG-based visualization framework for modeling languages defined by metamodeling techniques. Our framework combines XML standards with existing graph transformation and drawing technologies in order to provide an open, tool-independent architecture.

A dolgozatban bemutatunk egy automatikus, SVG alapú, metamodeljük által definiált modellek megjelenítésére szolgáló keretrendszert. Az általunk kidolgozott keretrendszer XML szabványokat és létező gráfrajzoló módszereket, eszközöket használ a nyílt, eszközfüggetlen architektúra érdekében.

Bevezetés

Napjainkban az **MDA (Model Driven Architecture - modellvezérelt architektúra)** a szoftverfejlesztés legmodernebb megközelítése. Eszerint a szoftver fejlesztése során először egy *platform-független modell* (PIM, Platform-Independent Model) készül el, amely a rendszer konkrét megvalósítástól független jellemzőit tartalmazza, melyből a konkrét megvalósítási környezet (szoftver architektúra) paramétereit is magában foglaló *platform-specifikus modell* (PSM, Platform-Specific Model) *automatikus modelltranszformációk* segítségével származtathatók. Végezetül a futtatható alkalmazás elkészítését a PSM modelleket feldolgozó *hatékony kódgenerátorok* támogatják. A fejlesztési folyamat során tehát olyan modellezési technológiára van szükség, amely segítségével mind a modellezési nyelvek, mind a különféle felhasználói modellek egységesen és precízen definiálhatók.

Egy *modellezési nyelv precíz leírásához* szükség van annak *absztrakt* (vagy logikai) *szintaxisának* definiálására, azaz tudnunk kell, hogy az adott nyelv milyen fogalmkészlettel rendelkezik, és az alapfogalmak hogyan kapcsolódnak egymáshoz (pl. a Petri-hálók modellezési nyelve esetén helyek, tranzíciók és ezek összeköttetései). Munkája során a tervező azonban nem az absztrakt, hanem a *konkrét szintaxisal* dolgozik (azaz pl. Petri-hálók esetén a helyeket mint köröket szeretné látni). Ezeket a *jólformáltsági kényszerekkel* kiegészítve (pl. egy helyet és egy tranzíciót összekötő él pontosan egy helyhez és egy tranzícióhoz tartozik) már precízen definiáltnak tekinthetjük a modellek *statikus struktúráját*, de természetesen szükséges még a *dinamikus szemantika* (hogyan működik a Petri háló?) megadása is.

Az **UML (Unified Modeling Language)**, amely az MDA-ban a közös modellezési nyelv szerepét töltené be, sajnos nem rendelkezik precíz szemantikával. A modellezési nyelvek precíz definíciójának fő irányvonalát a metamodellező eszközök képviselik (pl. VIATRA [1]). Ezek először egy tetszőleges modellezési nyelv absztrakt szintaxisát definiálják egy grafikus, UML-alapú *metamodellben*; majd az itt definiált elemkészlet segítségével építik fel a konkrét modelleket. Ezzel a módszerrel mind mérnöki mind matematikai modellezési nyelvek (pl. Petri hálók) statikus struktúráját is le lehet írni.

A rendelkezésre álló modellező eszközök közötti adatcserét lehetővé tévő közös, szabványos formátum az **XML (eXtensible Markup Language)**. Ez a megoldás kiforrottnak tekinthető az absztrakt szintaxis, azaz a modellek logikai felépítésének tárolása szempontjából (lásd PNML [2], GXL & GTXL [3]), de a modellek grafikus képének reprezentációja nem kellően megoldott.

Vektorgrafikus ábrák leírására létezik egy XML alapú nyelv, az **SVG (Scalable Vector Graphics - skálázható vektorgrafika)** [4], amely olyan alacsonyszintű grafikus primitívekből építkezik, mint pl. vonalak, körök, téglalapok; így, bár a megjelenítendő modellek grafikus képét le lehet írni vele, „kézi” tervezésre alkalmatlan.

A dolgozatban egy olyan megjelenítő keretrendszer vázát mutatjuk be (további információ: [5]), amely a metamodellel definiált modellezési nyelveknek megfelelő logikai modelleket képes egy metamodell szintű terv alapján automatizáltan és nyílt, eszközfüggetlen módon grafikusan megjeleníteni. Keretrendszerünkben a grafikus reprezentáció automatikus szintézisére gráfranzformációs ill. gráfrajzoló technológiákat használtunk föl, míg keretrendszerünk eszközfüggetlenségét XML alapú kapcsolódási pontok (interfészek) kialakításával értük el.

A megjelenítő keretrendszer

Az 1-es ábrán látható az általunk javasolt megjelenítő keretrendszer sematikus felépítése. Munkánk fő gondolatait a jól ismert Petri-hálók grafikus modellezési nyelvének példáján keresztül mutatjuk be, de a keretrendszer természetesen tetszőleges más modellezési nyelv esetén is alkalmazható.

1. ábra Az automatikus megjelenítési folyamat

Célunk egy konkrét Petri-háló modell transzformálása SVG-be, amely a legtöbb Internet böngészővel megnyitható. Ehhez a következő bemenetekre van szükségünk:

- **Petri-háló metamodell.** E metamodell megadja a Petri-hálók absztrakt szintaxisát, bevezeti pl. a helyek, tranzíciók, tökének fogalmát.
- **Petri-háló modell.** Természetesen szükség van a felhasználó megjelenítendő modelljére.
- **Az absztrakt és a konkrét szintaxis közötti leképezés** (transzformációs szabályok). Szükség van egy, a tervező által definiált modelltranszformációs szabályhalmazra, amely megadja, hogyan kell megjeleníteni egy Petri-hálót (pl. a helyeket körökként reprezentáljuk). Ezt a szabályhalmazt minden megjelenítendő modellezési nyelvhez egyszer kell elkészíteni, utána az alkalmazható lesz az adott nyelv tetszőleges modelljére.

Magát a megjelenítési folyamatot a következő jó i elkülönülő lépésekre bontottuk:

- **A logikai felépítés transzformálása rajzolási primitívekké.** Első lépésként a megjelenítendő modellt egy logikai rajzzá transzformáljuk, amely a modellezési nyelv elemei (helyek, tranzíciók) helyett már rajzolási primitíveket tartalmaznak (körök, téglalapok). A modellezési nyelv elemei közötti kapcsolatokat (pl. helyek és tranzíciók között futó élek) természetesen át kell menteni a rajzra is. Az eredmény egy logikai rajz, amely grafikai elemeket tartalmaz, de ezeknek még nincsenek fizikai koordinátáik, csak az egymáshoz való viszonyuk ismert (pl. egy nyíl egy körből indul és egy téglalapban végződik). Ezen transzformáció végrehajtására a tanszékünkön fejlesztett VIATRA [1] rendszert használjuk.
- **Koordináta-számítási lépés.** Az előző lépésben kapott logikai rajz elemeihez koordinátákat kell rendelni. Ezt a feladatot jelenleg az IBM GFC (Graph Foundation Classes) [6] nevű Java alapú gráfrajzoló osztálykönyvtára látja el.
- **Az SVG reprezentáció létrehozása.** Ennél a lépésnél már minden szükséges információ megvan a modell megjelenítésére, azonban még a rajzolási metamodellnek megfelelő formában. Ezen a ponton kell ebből a formátumból egy SVG reprezentációt előállítani. Ennek végrehajtására XSLT-t használunk (eXtensible Stylesheet Language for Transformations) [7]. Habár ez egy másik transzformációs technika, mint az előző (modelltranszformáció), azonban ez nem okoz gondot, hiszen ezt két, előre definiált nyelv között kell végrehajtani, így ez közös minden modellezési nyelv esetén.

A megjelenítési folyamat fentebb felvázolt lépései egymástól jól elkülönülnek, és feltéve, hogy megtartjuk a köztük definiált, szabványos XML interfészt, a három lépést végrehajtó eszközök bármelyike lecserélhető másikkra.

Összegzés

A dolgozatban metamodell segítségével definiált grafikus modellezési nyelvek modelljeinek automatikus megjelenítésére javasoltunk megoldást, mely módszertanilag elkülönülő, de szabványos XML interfészek által csatolt lépésekre bontható. A megjelenítési folyamat végeredménye egy (szintén XML alapú) SVG formátumú vektorgrafikus ábra.

További terveink között szerepel az általunk definiált rajzolási metamodell (melyet hely hiányában nem állt módunkban bemutatni) kibővítése, hogy komplex grafikus kényszereket is lehessen alkalmazni a konkrét szintaxis megadása során. Ehhez a CSVG (Constraint Scalable Vector Graphics) [8] nyelvet szeretnénk felhasználni, amely segítségével SVG primitívekre adhatunk meg különböző kényszereket lehet matematikai formában.

Hivatkozások

- [1] D. Varró, G. Varró, A. Pataricza: *Designing the automatic transformation of visual languages*, megjelenés helye: *Science of Computer Programming* 44 (2002.), 205-227. oldal
- [2] „Petri Net Markup Language”, URL: <http://www.informatik.hu-berlin.de/top/pnml>
- [3] G. Taentzer: *Towards common exchange formats for graphs and graph transformation systems*, megjelenés helye: UNIGRA 2001., *Uniform Approaches to Graphical Process Specification Techniques*, ENTCS 44(4), 2001.
- [4] World Wide Web Consortium, „Scalable Vector Graphics (SVG) 1.0 Specification”, 2001., URL: <http://www.w3.org/TR/SVG>
- [5] P. Domokos, D. Varró: *An open visualization framework for metamodel-based modeling languages*, megjelenés helye: GraBaTs 2002., *International Workshop on Graph-Based Tools*, Oct. 7-8., Barcelona, Spain, ENTCS 72 (2002), 78-87. oldal
- [6] IBM, „Graph Foundation Classes for Java”, URL: <http://www.alphaworks.ibm.com/tech/gfc>
- [7] World Wide Web Consortium, „XSL Transformations Version 1.0”, URL: <http://www.w3.org/TR/xslt>
- [8] G. J. Badros, J. J. Tirtowidjojo, K. Marriot, B. Meyer, W. Portnoy, A. Borning: *A constraint extension to Scalable Vector Graphics*, School of Computer Science and Software Engineering, Monash Univ., Dept. of Computer Science and Engineering, Univ. of Washington. URL: <http://www.cs.washington.edu/research/constraints/web/csvg-wwwIO>

Domokos Péter

V. éves egyetemi hallgató

e-mail: pdomokos@mit.bme.hu

Varró Dániel

doktorandusz

e-mail: varro@mit.bme.hu

Méréstechnika és Információs Rendszerek Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem
H-1521, Budapest,
Magyar tudósok körútja 2.
fax: +36 1 463 2667