DISTRIBUTED SIMULATION OF DEVS

Michal CERHÁK, Master Degree Programme (5)
Dept. of Intelligent Systems, FIT, BUT
E-mail: xcerha00@stud.fit.vutbr.cz

Supervised by: Ing. Vladimír Janoušek

ABSTRACT

This text aims on distributed approach to modeling and simulation based on the DEVS formalism. It describes some basic principles of the DEVS formalism such as atomic DEVS and coupled DEVS. It shows possibilities of creating hierarchical models and shows that DEVS gives us means to construct models which are reusable and interoperable and if it is useful it also allows us to use multiple formalisms for the model construction. Two basic approaches for controlling a distributed simulation are shown. Possibilities of usage of the simulator are explained. Summary, some technical details about the simulator and future plans can be found at the end of the paper.

1 ÚVOD

Tvorba simulačního modelu a jeho následná simulace obvykle probíhají na jednom stroji, v jednom procesu. Existují ale i případy, kdy je vhodné nebo dokonce nutné přistoupit k simulaci modelu v několika oddělených procesech nebo fyzicky na několika nezávislých strojích. Takovou simulaci nazýváme paralelní nebo distribuovanou. Modely simulované distribuovaně se typicky vyznačují vysokou výpočetní náročností. Jiným důvodem pro distribuovanou simulaci může být prostorové oddělení částí modelů.

Pro tvorbu simulačních modelů je výhodné a někdy nutné, aby součásti nebo i celé modely byly znovupoužitelné a měly přesně definované rozhraní komunikace s okolím. Tento požadavek zaručuje relativně jednoduché propojení již existujících modelů, které mohly být použity k jiným simulacím. Z těchto důvodů je jako formalismus pro tvorbu abstraktních modelů použit DEVS.

2 DISCRETE EVENT SYSTEM SPECIFICATION (DEVS)

Autorem formalismu DEVS je prof. Bernard P. Zeigler působící na University of Arizona. Autorem referenční implementace simulátoru je rovněž skupina prof. Zeiglera [1]. Formalismus DEVS slouží k tvorbě abstraktních simulačních modelů. Umožňuje vytvářet hierarchické modely s jednotným komunikačním rozhraním. Jak již název napovídá, simulace modelu založeného na DEVS je řízená událostmi [2]. Rozeznáváme dva druhy komponent

modelu: Atomický a spojovaný DEVS.

Atomický DEVS je nejmenší jednotka provádění modelu. S vnějším prostředím komunikuje pomocí vstupních a výstupních portů. Uvnitř se nachází stavové řízení, jehož složitost není nijak předepsána. Z toho plyne, že může obsahovat i modely jiných formalismů, od konečných automatů, přes Petriho sítě, až po algoritmy spadající do třídy Turingových strojů. Tato vlastnost poskytuje velkou flexibilitu při tvorbě abstraktních simulačních modelů založených na DEVS a umožňuje tvorbu heterogenních modelů.

Spojovaný DEVS obsahuje síť vnořených atomických nebo spojovaných komponent. Vnořené komponenty jsou propojeny pomocí vstupních a výstupních portů. Tento mechanismus umožňuje tvorbu hierarchických modelů. Spojované komponenty se v průběhu simulace starají o správné rozesílání řídících zpráv. Z hlediska topologie modelu rozeznáváme tři druhy zpráv. Vstupní zprávy přicházejí na vstupní porty spojovaného DEVSu a jsou dále rozesílány na vstupní porty vnořených modelů. Vnitřní zprávy produkují vnořené modely. Jsou zasílány přes vnitřní propojení na vstupní porty jiných vnořených modelů. Výstupní zprávy opět generují vnořené modely, ale na rozdíl od vnitřních zpráv jsou směrovány na výstupní porty spojovaného DEVSu.

Aby bylo dosaženo kompatibility atomických a spojovaných komponent, musí být stanoveno jednotné komunikační rozhraní. Každá komponenta musí rozumět právě čtyřem zprávám [2]: Inicializace modelu, příchod vnější události, naplánovaný interní přechod a generování výstupních dat.

3 DISTRIBUOVANÁ SIMULACE

Z hlediska modelu by měla distribuovaná simulace probíhat stejně jako běžná simulace prováděná v jednom procesu. Veškerá zodpovědnost za korektní průběh simulace je tedy převedena na simulátor. Největším problémem z hlediska provedení simulace je zachování kauzality. To znamená, že událost E_i, ovlivňující událost E_j, musí být provedena dříve nezávisle na tom, zda obě události proběhnou v jednom simulátoru či nikoliv.

V sekvenční simulaci není složité dodržet pořadí provádění událostí. Stačí například vybrat události s minimální časovou známkou a ty provést. V distribuovaném prostředí se tento úkol stává problematickým. Důvodem je to, že dílčí simulátory mají informace pouze o té části modelu, kterou právě simulují. Pro řešení zachování kauzality u distribuovaných simulátorů existují dva základní přístupy: Konzervativní a optimistický [2].

Konzervativní přístup je založen na striktním zachování kauzality ve všech dílčích simulátorech v každém okamžiku simulace. Z tohoto požadavku vyplývá potřeba synchronizace simulátorů při komunikaci vzdálených komponent. Synchronizace je realizována speciálním druhem synchronizačních zpráv [2].

Optimistický přístup, na rozdíl od konzervativního, dovoluje dočasné porušení kauzality. Při detekci porušení kauzality je simulace vrácena do stavu před tímto porušením. To má za následek stornování všech vygenerovaných zpráv a propagaci tohoto porušení kauzality do dalších závislých komponent.

4 VYUŽITÍ DISTRIBUOVANÉHO SIMULÁTORU

Simulátor bude konstruován tak, aby bylo možné za běhu simulace migrovat jednotlivé komponenty modelu ze stávajícího simulačního serveru na jiný. Tato vlastnost má několik možností využití.

První z nich je automatické vyvažování zátěže jednotlivých simulačních serverů. To může být velmi užitečné pro zajištění plynulosti simulace. Pokud by na jednom serveru běželo několik náročnějších výpočtů, byla by celá simulace brzděna právě tímto serverem.

Další možností je simulace mobilních agentů. Každý simulační server by představoval část prostředí a agent by se v tomto prostředí fyzicky pohyboval.

V neposlední řadě může být migrace komponent využita v model-based designu. Zpočátku se bude simulovat nějaký reálný systém. V průběhu simulace můžeme dostat k dispozici libovolnou část tohoto reálného systému. Komponentu modelu, která bude odpovídat této reálné části systému přesuneme na jiný simulační server a tam provedeme záměnu modelu za skutečnou část. To bude možné provést za běhu simulace, takže tím běžící simulaci nijak neporušíme a navíc můžeme testovat chování reálného zařízení.

Díky možnosti migrace komponent lze také snadno přejít od jednoduché simulace k simulaci distribuované.

Další rozšíření simulátoru poskytuje prostředky pro přechod k simulaci v reálném čase a pro interaktivní zásahy v průběhu simulace.

5 ZÁVĚR

Simulátor bude realizován pomocí konzervativního přístupu řízení simulace. Toto řešení je méně náročné na implementaci, ale v určitých případech může více zatěžovat komunikační médium. Pro komunikaci mezi jednotlivými simulátory byl zvolen Ethernet, jenž by měl zajišťovat dostatečnou kapacitu pro přenos všech potřebných informací mezi jednotlivými simulačními servery. V současné době byla provedena analýza a návrh simulačního serveru. Byla realizována podpora serializace DEVS komponent, prostředky pro síťovou komunikaci a část komunikačního protokolu.

Význam tohoto simulátoru spočívá v možnosti dynamických změn součástí modelu, ale i v možnosti změn topologie celého modelu. Seznam dostupných implementací simulátorů využívajících formalismus DEVS je dostupný na [3].

LITERATURA

[1] DEVSJAVA, webová stránka věnovaná simulátoru. Dokument dostupný na URL http://www.acims.arizona.edu/SOFTWARE/software.shtml#DEVSJAVA (duben 2006)

[2] Zeigler, B., P., Praehofer, H., Kim, T., G.: Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems, San Diego, California, USA, Academic Press 2000, ISBN 0-12-778455-1

[3] DEVS tools, seznam implementací DEVS. Dokument dostupný na URL http://www.sce.carleton.ca/faculty/wainer/standard/ (duben 2006)