SCIENTIFIC PAPERS
OF THE UNIVERSITY OF PARDUBICE
Series B
The Jan Perner Transport Faculty
7 (2001)

APLIKACE PARADIGMATU AUTONOMNÍCH AGENTÙ NA ARCHITEKTURU SIMULAÈNÍHO MODELU

Antonín KAVIÈKA

Katedra informatiky v dopravì

1. Úvod

V procesu zkoumání, pøestavby a projektování reálných systémù jsou uplatòovány simulaèní modely, které umožòují v rámci experimentù testovat rùzné varianty provozu daných systémù a na základì jejich výsledkù je možné pøijmout vhodná opatøení, která umožní napø:

- racionálnìjší provoz stávajících systémù,
- efektivní provoz budoucích tj. projektovaných systémù a
- pøípustný provoz stávajících systémù v rùzných fázích jejich rekonstrukce resp. pøestavby.

Výše zmínìná opatøení, která se aplikují na reálný systém mohou být reprezentována napøíklad: úèelnými modifikacemi technologických procesù a s tím spojeným snížením poètu lidských zdrojù, modernizací resp. dostavbou technických zaøízení nebo dopravní infrastruktury sytému apod.

V posledních letech je tendence budovat simulaèní modely stále komplexnìjších systémù, a proto je nutné, aby i samotná metodika tvorby simulaèního modelu jakož i

jeho architektura vyhovovala souèasným požadavkùm. Z tohoto dùvodu se výzkum vtéto oblasti zamì øuje na takové pøístupy, které dovolují zejména:

- transparentní a dobøe srozumitelnou organizaèní strukturu (napø hierarchii) øídících jednotek simulaèního modelu,
- osamostatnì ní a modularizaci øídících jednotek modelu,
- integraci èlovì ka do simulaèního modelu a jeho spolupráci s modelem v rámci interaktivního režimu,
- konfigurování a parametrizaci modelu bez nutnosti zasahovat do programového kódu programu.

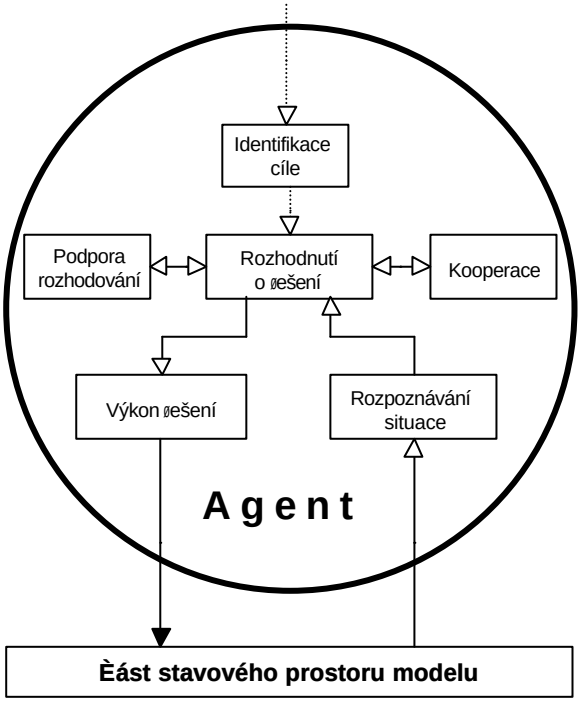
Modelované systémy jsou z hlediska rozhodování a øízení vì tšinou organizované do hierarchicky strukturovaných komponentù s vymezenou kompetencí rozhodování. Do rùzných úrovní hierarchie se soustøeï ují rùzné úrovnì rozhodování (napø strategické, taktické, operativní). Na jedné úrovni hierarchie jsou komponenty oprávnì né rozhodovat v urèité èásti reálného svì ta (napø správce technických zaøízení, správce mobilních obslužných zdrojù, dispeèer apod.). Modely zmínì ných komponentù se èasto oznaèují pojmem agent. Agent se v oblasti své kompetence soustøeï uje na všechny funkce dùležité pro øízení (analýza cíle – rozpoznávání situace – výkon vyplývající z pøijatého rozhodnutí), je schopný pracovat autonomnì a spolupracovat pøi øešení problémù s jinými agenty.

2. Agent a jeho komponenty

Pod pojmem agent resp. autonomní agent rozumíme [2]: zapouzdøený poèítaèový systém zasazený do nì jakého okolí, který v nì m pružnì a autonomnì pùsobí za úèelem plnì ní daného cíle, pøièemž za jeho klíèové vlastnosti považujeme [8]:

- autonomnost, tj. agent je schopen pracovat samostatnì bez vnì jších intervencí a zcela øídit své výkony a kontrolovat svùj vnitøní stav,
- spoleèenské chování, které se projevuje jako interakce s jinými agenty (resp. s èlovì kem) prostøednictvím jistého komunikaèního mechanismu/jazyka,
- reaktivitu neboli reagování na podnì ty z okolí a
- iniciativnost tzn. že agent nereaguje pouze na podnì ty z okolí, nýbrž je schopen cíleného chování vyvíjením vlastní iniciativy.

Na základì výše uvedeného tedy lze vyspecifikovat hlavní funkce agenta, které jsou uvedeny na obrázku *obr. 1*. Po identifikaci cíle øídí agent „svùj“ úsek umì lého svì ta (stavového prostoru simulaèního modelu) v cyklu: rozpoznávání situace – rozhodování o øešení – výkon øešení. Pro podporu rozhodování mu slouží funkce návrhu øešení problémù a kooperace s jinými agenty (resp. s èlovì kem). Rozpozná-li situaci, jejíž øešení není v jeho kompetenci, pomáhá øešit problémy jiným agentùm tím, že je o dané situaci uvì domí.

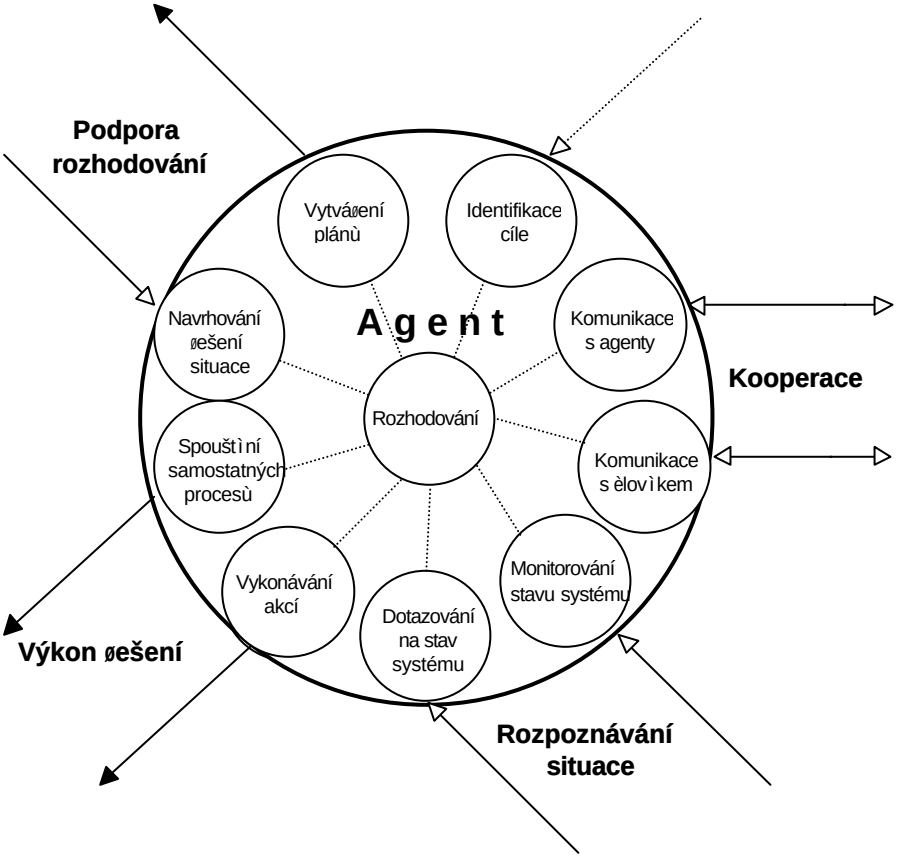


Obr. 1 *Funkce agenta*
Fig. 1 *Function of agent*

Na obrázku *obr. 2* jsou popsány schopnosti agenta na podrobnìjší úrovni. Pro funkci kooperace agenta slouží jeho schopnost komunikovat s okolím (jinými agenty resp. èlovìkem). Pro podporu rozhodování disponuje schopností navrhovat jednorázová øešení problémù (typicky volání optimalizaèních algoritmù), jakož i schopnost vytváøení dlouhodobìjších plánù øešení problémù. Percepèní (senzorickou) funkci agenta zabezpeèuje schopnost jednorázovì se dotazovat na stav systému nebo spustit „monitor“, který iniciativnì a samostatnì informuje o jistých situacích. A koneènì výkonnou funkci naplòuje buï aktivací akce, která okamžitì zmìní stav systému anebo spuštìním samostatného procesu, který potom bez dalších vnìjších zásahù mìní stav sytému v jistých èasových okamžicích.

Schopnosti agenta je vhodné rozdìlit do skupin a každou takovouto schopnost implementovat jako samostatný interní komponent. Hlavními dùvody pro zmínìnou dekompozici agenta jsou:

- možnost sdílení (využívání) komponentu vícero agenty,
- možnost implementace více alternativních verzí jednoho komponentu, znichž si uživatel vybere jeden vhodný pøi sestavování scénáøe simulaèního experimentu.



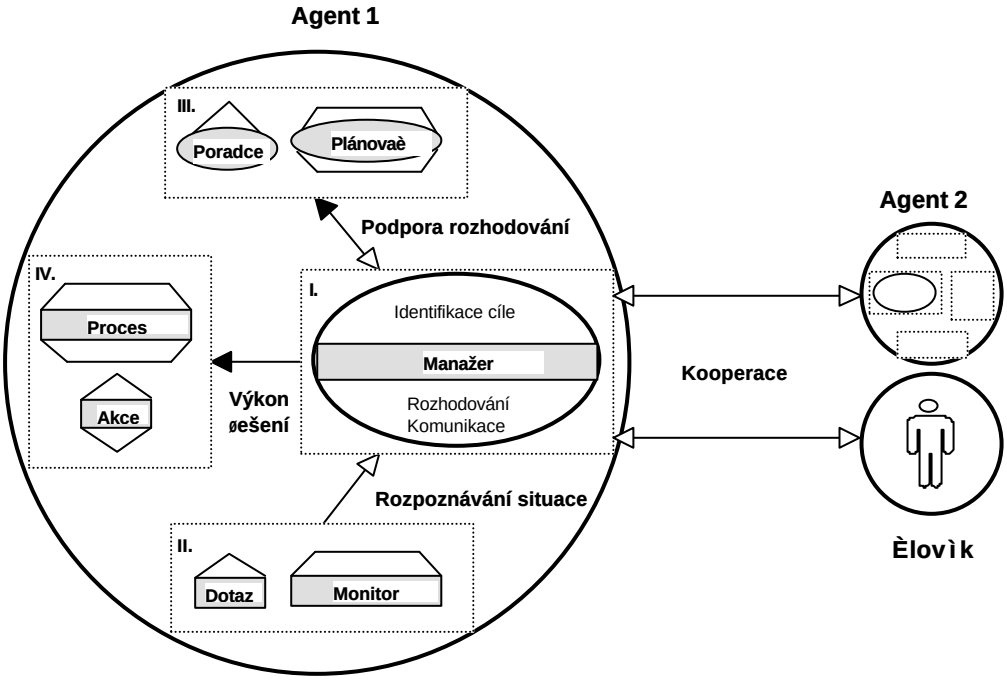
Obr. 2 *Schopnosti agenta*
Fig. 2 *Posibility of agent*

Agenta dekomponujeme (*obr. 3*) na èetyø skupiny komponentù. První skupina øídících a rozhodovacích komponentù obsahuje jediný druh komponentu zvaný manažer, který ze schopností agenta pøebírá schopnost identifikace cíle, komunikace a rozhodování. Manažer pøedstavuje ústøední komponent v tom smyslu, že dokáže komunikovat s ostatními interními komponenty (ty mezi sebou nekomunikují).

Zpøístupnìní informací o stavu systému zabezpeèuje skupina informátorù, která mùže obsahovat komponenty dvou druhù. Dotaz zprostøedkuje informace na pokyn manažera okamžitì, naproti tomu monitor zkoumá opakovanì po svém spuštìní stav systému v jistých èasových okamžicích ze zvoleného aspektu a zprostøedkuje manažerovi ty informace, které jsou pro nìj významné.

Skupina øešitelù podporuje rozhodnutí manažera tím, že mu poskytuje návrhy øešení problémù. Poradce je pasivní komponent, který pouze na pokyn manažera okamžitì navrhne zpùsob (zpùsoby) øešení problému. Poradcem je obvykle optimalizaèní algoritmus nebo èlovìk. Jiným zpùsobem prezentování návrhù øešení problémù je

predikce jejich vzniku, kdy jsou vpøedstihu pro jistý èasový interval navržena øešení více problémù (napø èasové plány pøidìlování prostøedkù obsluhy). Komponent plánovaè mùže vytváøet takovéto plány øešení a sám je ve zvolených èasových okamžicích aktualizovat bez iniciativy manažera.



Obr. 3 *Dekompozice agenta*

Fig. 3 *Decompozition of agent*

Poslední skupinou komponentù jsou exekutoøi, kteøí vykonávají rozhodnutí manažerù o zmìnì stavu systému. Žádné jiné komponenty nemohou mìnit stav systému. Akce zabezpeèí jednorázovou a okamžitou zmìnu stavu (napø zvìtšení napìtí, pøepnutí výhybky apod.), zatímco proces po svém spuštìní mìní autonomnì hodnoty stavových promìnných vjistých èasových okamžicích (napø pøepíná stavy signalizaèních zaøízení na køižovatce).

Exekutory, informátory a øešitele souhrnnì nazýváme efektory, pøièemž tyto je možno dìlit na:

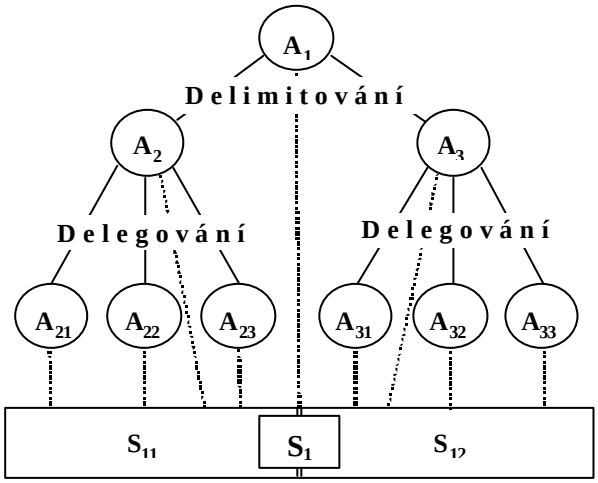
- èasové efektory, jejichž aktivita trvá nenulový simulaèní èas (procesy, monitory a plánovaèi) a
- promptní efektory (akce, dotazy a poradci), které jsou vykonány vjednom okamžiku simulaèního èasu.

Architekturu simulaèního modelu založenou na výše zmínìných „základních stavebních kamenech“ – autonomních agentech též zkrácenì oznaèujeme jako ABAsim (Agent-based Architechture of simulation model).

3. Multiagentový pøístup

Pro jednoduché systémy by se mohl simulaèní model skládat z jediného agenta. Avšak pøi modelování komplexních reálných systémù potøebujeme napø vìrnì zachytit organizaci (obvykle hierarchii) øídících prvkù, kterým mohou v simulaèním modelu odpovídat jednotliví agenti pracující též v urèité organizaci a mající jisté subordinaèní a komunikaèní vazby. Mluvíme tedy o tzv. mutltiagentovém pøístupu tj. o používání agentù v jistých organizaèních strukturách s definovanými vazbami.

Na obrázku *obr. 4* je znázornìn pøíklad, kdy model sestává z hierarchie kooperujících agentù, kde $\mathbf{A_1}$ (agent dopravní sítì) rozdìluje celou sí• $\mathbf{S_1}$ na dva regiony $\mathbf{S_{11}}$ a $\mathbf{S_{12}}$ a *delimituje* jejich správu na dva „regionální“ agenty stejného typu $\mathbf{A_2}$ a $\mathbf{A_3}$, pøièemž si ponechává koordinaci jejich práce.



Obr. 4 *Hierarchická struktura agentù*
Fig. 4 *Hierarchical structure of agents*

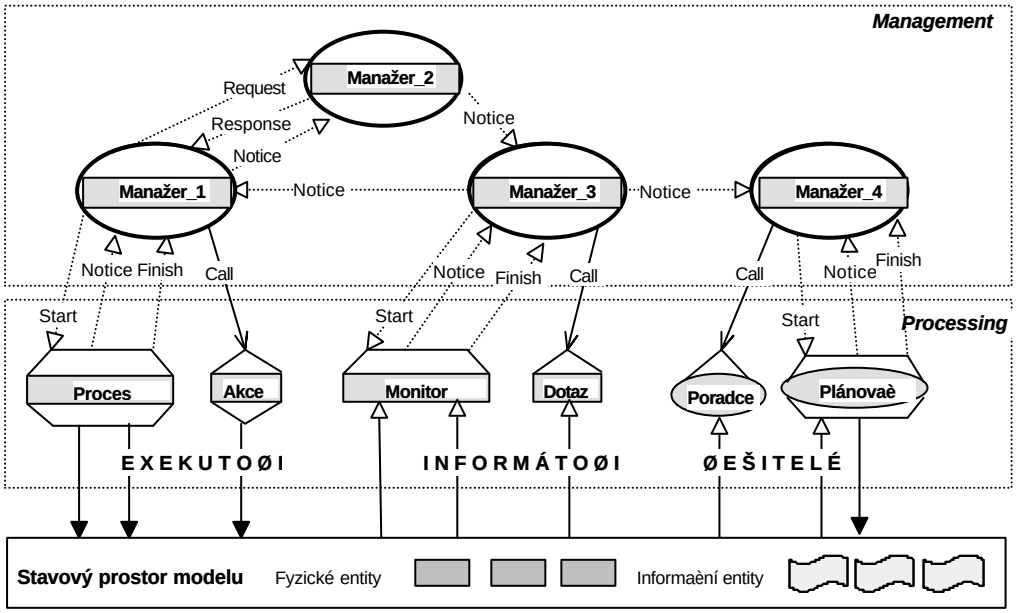
Každý z regionálních agentù používá k plnìní svých cílù (úkolù) další úžeji specializované agenty na nì ž *deleguje* èást svých kompetencí.

4. Vrstvový MPE-model

Dekompozice agentù na jednotlivé interní komponenty dvou základních skupin manažery a efektory nás pøivádí k myšlence podívat se na celý simulaèní model jako na systém složený ze dvou vrstev (*obr. 5*):

- *vrstvy managementu*, v níž jsou pøijímána rozhodnutí a formulovány pokyny k jejich realizaci a

- *vrstvy zpracování* (processing), jejíž komponenty realizují požadované výkony v rámci stavového prostoru simulaèního modelu.



Obr. 5 *Vrstvy simulaèního modelu*
Fig. 5 *Echelons of simulator model*

Kromì dvou výše zmínì ných vrstev (obsahujících interní komponenty agentù) mùžeme uvažovat ještì jednu vrstvu - *vrstvu entit*, která je souèástí stavového prostoru simulaèního modelu. Ta obsahuje fyzické a informaèní entity. *Fyzické entity* jsou modely jednotlivých prvkù modelovaného systému. Souhrn všech fyzických entit je datová struktura, která je statickým modelem stavu modelovaného systému v aktuálním okamžiku simulaèního èasu. *Informaèní entity* soustøeï ují informace, které nejsou pøímo souèástí fyzických entit (nepøedstavují jejich atributy), ale jsou buï potøebné k øízení systému (technologické plány, báze znalostí) nebo jsou to informace popisující chování modelu bì hem simulace (napø statistické údaje).

Z hlediska prùbì hu simulace je nyní zøejmé, že úkolem manažerù je ve vzájemné kooperaci a za pomoci informátorù a øešitelù startovat èinnost exekutorù ve správném èase a pøi splnì ní potøebných podmínek.

V popisované architektuøe je komunikace mezi manažery, jakož i mezi manažery a efektory realizována výhradnì pomocí zasílání zpráv (z toho pohledu mùžeme tuto architekturu též oznaèit jako *zprávovì-orientovanou*), které mùže klasifikovat následovnì :

Manažer je schopen zaslat zprávy typu *Start* resp. *Break* libovolnému èasovému efektoru, pøièemž po pøijetí této zprávy tento zahájí resp. pøeruší svoji èinnost.

Pro komunikaci mezi agenty jsou používány následující tøi typy zpráv:

- *Notice* (oznámení), pomocí níž je odesílána informace, aniž by byla oèekávána odpovì ï ,
- *Request* (žádost) obsahující urèitou žádost, pøièemž na ni odesilatel oèekává od adresáta odpovì ï ,
- *Response* (reakce/odezva), která reprezentuje odpovì ï na zprávu typu Request, pøièemž mùže být zaslána pouze jejímu pùvodci.

Každý èasový efektor povinnì zasílá v okamžiku ukonèení své èinnosti zprávu *Finish* „svému manažerovi“ tj. manažerovi, který jej spustil. Dále mohou èasové efektory bì hem své èinnosti v libovolném okamžiku zasílat zprávy typu:

- *Notice* (s urèitým oznámením) svému manažerovi a
- *Hold*, která pøedstavuje zprávu s *èasovým razítkem* (èasovým údajem), který definuje èas doruèení zprávy. Tento typ zprávy si sèasovým zpoždì ním zasílá èasový efektor sám sobì , tedy mùže po odeslání této zprávy opì t pokraèovat ve své èinnosti až od toho okamžiku simulaèního èasu kdy je mu zmínì ná zpráva doruèena (do té doby je neaktivní).

Zdùraznì me, že vpopisované architektuøe je realizováno plynutí simulaèního èasu pomocí zasílání zpráv pouze jediného druhu (Hold) a tedy èasová synchronizace modelu je koncentrována vèasových efektorech. Ve vrstvì managementu se zprávy sèasovými razítky nepoužívají a tudíž zde „neplyne“ simulaèní èas.

5. Metodika návrhu struktury „agentového“ modelu

Vì nujme se nyní struènì doporuèené metodice návrhu/tvorby struktury simulaèního modelu (postaveného na architektuøe kooperujících autonomních agentù) s jehož pomocí máme vyøešit daný konkrétní úkol.

V první fázi provádíme analýzu reálného (modelovaného) sytému, na nì mž si vymezíme *objekt zkoumání.* Dále si na objektu zkoumání vymezíme *systém* (sestávající ze struktury jistých prvkù), který odráží dílèí pøedmì t našeho zájmu a jehož prvky zahrnují pouze významné vlastnosti odpovídajících prvkù reálného systému (z hlediska øešení našeho úkolu).

V rámci druhé fáze navrhujeme jednotlivé specializované agenty, kteøí jsou kompetentní pro øešení definovaných problémù a operují nad vymezenými (a obvykle komplementárními) èástmi stavového prostoru simulaèního modelu.

Další fáze se vì nuje vytvoøení vzájemných vazeb mezi jednotlivými agenty z hlediska jejich organizace/hierarchie, kooperace a komunikace.

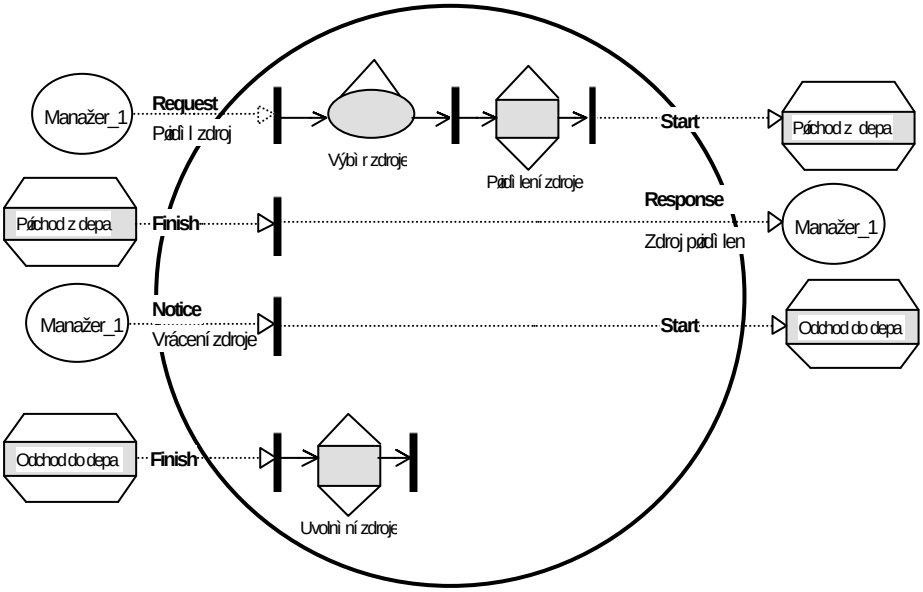
Koneènì v poslední fázi návrhu struktury modelu hodnotíme vhodnost celé navržené struktury z implementaèního hlediska tj. posuzujeme rùzné varianty možného sdílení interních komponentù agentù apod.

Po ukonèení uvedeného návrhu struktury simulaèního modelu následuje standardní „postavení, provìøení a používání/spouštìní“ simulaèního modelu tzn. že daný model je implementován, dále je provedena jeho verifikace a validace a po té je možné model využívat pro realizace sérií vhodnì navrhovaných simulaèních experimentù s cílem vyøešit zadaný konkrétní úkol.

6. Poznámky k implementaci

Programová (simulaèní) podpora pro tvorbu simulaèních modelù v souladu s konceptem ABAsim byla vytvoøena v rámci prostøedí softwarového produktu Borland Delphi pomocí programovacího jazyka Object Pascal. Zmínìná podpora sestává zejména ze *simulaèního jádra* („motoru simulaèního modelu“), které implementuje mechanismus pro realizaci simulaèního bìhu (replikace) zabezpeèující synchronizaci doruèování zpráv jednotlivým komponentùm modelu. Simulaèní jádro umožòuje budovat jednak *diskrétní* simulaèní modely a jednak *kombinované diskrétnì-spojité* simulaèní modely [6]. Navíc simulaèní jádro podporuje animaci v prùbìhu simulaèního bìhu.

Další významnou èástí simulaèní podpory je *interpret* modifikovaných zprávovì-orientovaných Petriho sítí, které pøedstavují mechanismus pro formální popis èinnosti manažerù (viz. *obr. 6*).



Obr. 6 *Pøíklad formalizace popisu manažera „zdrojù obsluhy“*

Obr. 6 *Example of „source attendants“ description manager formalization*

„Mozek“ každého manažera tedy mùžeme definovat pomocí uvedeného formalismu, pøièemž pøi samotném vývoji simulaèního modelu je pøíslušný formální popis

zkonstruován v rámci specializovaného grafického editoru a je ve formì dat uložen do databáze manažerù. Pùsobení odpovídajícího manažera bì hem simulaèního bì hu je potom realizováno jako interpretace odpovídajících dat pøíslušné Petriho sítì. To tedy znamená, že èinnost øídících komponentù simulaèního modelu není definována pomocí kódu programu, nýbrž pomocí dat.

7. Simulaèní model železnièní stanice

Uveï me si významný pøíklad aplikace ABAsim architektury. V procesu racionalizace (a pøípadnì reorganizace a modernizace) technologie nákladní dopravy v rámci rùzných železnièních spoleèností vyvstává potøeba komplexnì zhodnotit provoz významných stanic (zejména seøaï ovacích) na železnièní síti. Je proto nutné sestrojit vhodný model provozu stanice a na nì m testovat jeho rùzné provozní varianty [4], pøièemž je možné sledovat dùsledky zmì n napøíklad infrastruktury, složení a poètu obslužných zdrojù, intenzit vstupních a výstupních proudù jakož i technologií lokálních obslužných procesù.

Pro studium tak komplexního systému, jakým je seøaï ovací nádraží je asi nejvhodnì jším modelem poèítaèový simulaèní model, který je dostateènì flexibilní vzhledem k potøebì otestování vì tšího poètu dosti odlišných provozních variant.

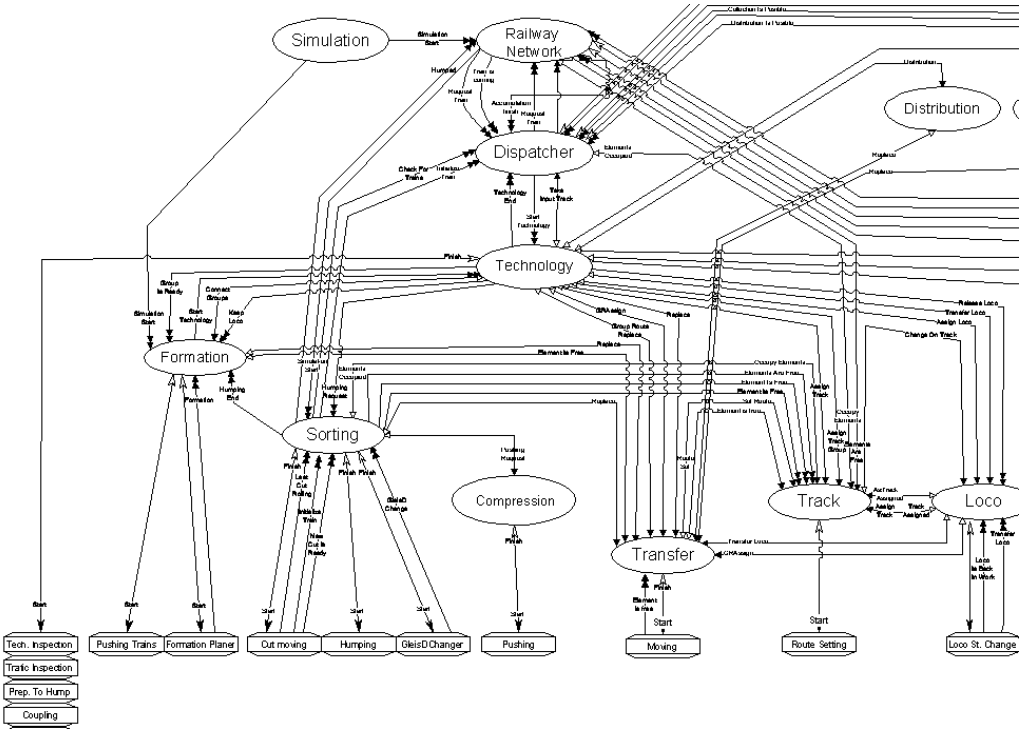
S touto motivací byl vyvinut komplexní simulaèní model železnièní seøaï ovací stanice [3], který je postaven v ABAsim architektuøe. Na obrázku *obr. 7* vidíme zjednodušenou èást jeho vrstvového MPE-modelu, z nì hož je patrno, z jakých specializovaných øídících komponentù se skládá (napø obsahuje manažera-dispeèera /dispatcher/, manažera obslužných technologií /technology/, manažera kolejištì /track/, tøídì ní /sorting/, stlaèování /compression/, manažera lokomotiv /loco/ atd.). Dále je vidì t, které øídící komponenty spouštì jí jednotlivé procesy (napø manažer tøídì ní spouští proces rozpouštì ní /humping/, manažer technologií proces technické prohlídky /technical inspection/ apod.).

Pro potøeby uživatelsky pøíjemného ovládání zmínì ného modelu byl vyvinut simulaèní nástroj *VirtuOS*, který pøedstavuje integrované prostøedí (experimentální laboratoø) v jehož rámci je možné pohodlnì vytváøet rùzné konfigurace modelu (rùzné provozní varianty) a provádì t série simulaèních experimentù (nabízejících též animaèní výstupy, které umožòují sledovat vývoj provozu v podobì blízké realitì) s jejich následným statistickým a grafickým vyhodnocením.

Uveï me alespoò nì které významnì jší projekty, vjejichž rámci byl nástroj VirtuOS (a tím aplikována i architektura ABAsim) úspì šnì použit:

- Projekt *Seøaïovací stanice Linec /Rakousko/* (Linz VBf) – simulaèní studie k posouzení rùzných variant pøestavby stanice Linec [5].
- Projekt *Hamburg Alte Süderelbe /Nìmecko/* – simulaèní studie k posouzení kapacity infrastruktury stanice Alte Süderelbe (na území mì sta Hamburg).

- Projekt *Centrální seøaïovací nádraž í Vídeò /Rakousko/* (Wien ZVBf) – simulaèní studie k posouzení rùzných provozních variant stanice.
- Projekt *Seøaïovací nádraž í Mudanjiang /Èína/* – simulaèní studie k posouzení rùzných provozních variant stanice.



***Obr. 7** MPE-model ž elezniční stanice*
***Fig. 7** MPE-model of railway station*

Simulaèní nástroj VirtuOS mùže mít samozøejmì i širší použití, tzn. že vjeho prostøedí mohou být na bázi ABAsim architektury budovány i simulaèní modely napø vleèek, osobních železnièních stanic, pøístavních pøekladiš• nebo uzlù intermodální dopravy (kontejnerové terminály). Pro tyto úèely je však nutné, aby byla struktura stávajícího simulaèního modelu seøaï ovací stanice modifikována resp. rozšíøena o takové komponenty, které budou odrážet specifika ostatních druhù zmínì ných dopravních uzlù.

8. Závìr

Na základì dosavadních zkušeností spoužíváním ABAsim architektury mùžeme závì rem zdùraznit zejména tyto její pøednosti:

- Struktura agentového modelu je velmi blízká struktuøe modelovaného systému zejména z pohledu organizace/hierarchie øídících jednotek. Tato vlastnost je

výhodou nejen pro samotného designera simulaèního modelu, ale i pro komunikaci se zadavatelem/zákazníkem, který je schopen snadno pochopit strukturu modelu a mùže upozornit ze svého pohledu na jeho korektnost.

- Modulárnost celého modelu a možnost používat jeho jednotlivé moduly i vdalších odlišných modelech resp. vmodelech modifikovaných. Stupeò modularizace vždy záleží na designerovi modelu, tzn. že jednotlivé moduly mohou zahrnovat napøíklad buï celé agenty (nebo jejich skupiny) anebo jejich jednotlivé komponenty (zejména øídící).
- Pøirozená integrace èlovìka do simulaèního modelu, který zde mùže být chápán buï jako autonomní agent nebo jako jeden z jeho komponentù (poradní, senzorický nebo øídící). Èlovìk tudíž mùže prostøednictvím speciálního obrazovkového interface komunikovat sostatními agenty tzn. pøijímat od nich resp. jim zasílat zprávy vrámci interaktivního/kooperativního režimu.
- Podpora vytváøení spíše univerzálnìjších a flexibilních simulaèních modelù než jednoúèelových modelù. Je pøirozené vytváøet si databázi alternativních komponentù resp. celých agentù, z nichž je možno vrámci definice scénáøe simulaèního bìhu „namíchat“ požadovanou verzi/alternativu modelu. Designer tedy mùže modifikovat:
  - výkonné vlastnosti agenta (výbìrem jeho exekutorù),
  - rozhodovací vlastnosti agenta (vybírajíc z množiny alternativních informátorù a øešitelù),
  - øídící strategie agenta (výbìrem z rùzných variant „mozkù“ manažerù) nebo
  - modelování celých èástí modelovaného systému (výbìrem celých alternativních agentù).
- Možnost, aby experimentátor mohl vytváøet konfigurace modelu a simulaèní scénáøe editaèními prostøedky bez nutnosti zasahovat do kódu programu.
- Podpora spojování modelù do sítí a tedy podpora budování modelù sítí a sítì modelù.
- Koncept zprávovì orientované architektury je pøedpokladem pro tvorbu distribuovaných simulaèních modelù (s možností realizace distribuované interaktivní simulace), a proto je uvedená architektura principiálnì pøipravena na svùj další vývoj a rozšíøení do distribuovaného prostøedí.

Použití architektury ABAsim lze doporuèit zejména pro tvorbu simulaèních modelù komplexních systémù, které lze vjejím rámci pøehlednì asrozumitelnì strukturovat a bez problémù rozšiøovat resp. modifikovat.

Lektoroval: Prof. Ing. Ladislav Skýva, DrSc. akademik

Pøedlož eno: v únoru 2002.

Literatura

1. Kavièka, A. *Architektura simulaèního modelu založ ená na kooperujících autonomních agentech*. Sborník pøíspì vků mezinárodní konference informaèních technologií vdopravì INFOTRANS 2002, Pardubice, 2002. ISBN 80-7194-419-X.
2. Jennings, N., R. *An agent-based approach for building complex software systems*. Communications of the ACM, April 2001, Vol. 44, No.4, pp.35-41.
3. Klima, V., Kavièka, A., Adamko, N. *Software tool VirtuOs – simulation of railway station operation*. Proceedings of European simulation multiconference, SCS, Prague, 2001, pp.84-88. ISBN 1-56555-225-3.
4. Klima, V., Kavièka, A. *Simulation support for railway infrastructure design and planning processes*. Proceedings of COMPRAIL 2000 conference in Bologna – Italy, Wessex Institute of Technology-Computational Mechanics Publications, Southampton-UK, September 2000, pp.447-456. ISBN 1-85312-826-0.
5. Kavièka, A., Klima, V., Niederkofler, A., Za•ko, M. *Simulation model of marshalling yard Linz Vbf (Austria).* Proceedings of The international workshop on Harbour, Maritime & Logistics Modelling and Simulation, SCS, Genoa, Italy, 1999, pp.317-320. ISBN 1-56555-175-3.
6. Kavièka, A., Klima, V., Adamko, N., Fabian, P. *System for combined simulations*. In: Proceedings of European Simulation Symposium & Exhibition - ESS '96, Society for Computer Simulation International, Genoa, Italy, 1996, vol.II, pp.240-244. ISBN 1-56555-099-4.
7. Klima, V., Kavièka, A. *Agent-based simulation model design*. Proceedings of European simulation multiconference, SCS, Budapest, 1996, pp.254-258. ISBN 1-56555-097-8.
8. Wooldridge, M., Jennings, N.,R. *Intelligent Agents: Theory and Practice*. The Knowledge Engineering Review 10, 2/1995, pp.115-192.

Resumé

APLIKACE PARADIGMATU AUTONOMNÍCH AGENTÙ NA ARCHITEKTURU SIMULAÈNÍHO MODELU

Antonín KAVIÈKA

Èlánek popisuje originální architekturu simulaèního modelu, která je založena na konceptu kooperujících autonomních agentù. Každý agent pøedstavuje samostatný øídící modul, jenž je zodpovì dný za „správu“ urèité èásti modelu (jeho stavového prostoru) a vpøípadì výskytu problému, který není schopen sám øešit, využívá kooperaci s ostatními agenty. Agenti sestávají ze specializovaných komponentù, které jsou zamì øeny na vykonávání jednotlivých funkcí agenta. Chování agenta je možné velmi pružnì modifikovat „výmì nou“ jeho požadovaného komponentu což pøíspívá kvysoké flexibilitì celého modelu. Zmínì ný koncept tvorby simulaèního modelu poskytuje øadu výhod z nichž za hlavní je možné považovat:

- podobnost struktury modelu struktuøe modelovaného systému,
- modulárnost systému a z toho vyplývající znovupoužitelnost jeho jednotlivých modulù,
- integrace èlovì ka jako jednoho z rozhodovacích, resp. poradních komponentù systému,
- podpora vytváøení spíše zobecnì ných než jednoúèelových simulaèních modelù,
- možnost, aby experimentátor mohl vytváøet konfigurace modelu a simulaèní scénáøe editaèními prostøedky, bez nutnosti zásahu do kódu programu,

- podpora spojování modelù do sítí, a tedy podpora budování modelù sítí a sítì modelù,

Agentová architektura simulaèního modelu byla s úspìchem aplikována pøi tvorbì komplexního modelu železnièního seøaïovacího nádraží a otestována v øadì zákaznických projektù, v jejichž rámci byly uplatnì ny všechny výše uvedené výhody zmínì né architektury.

Summary

THE APPLICATION OF AUTONOMOUS AGENT PARADIGM TO THE SIMULATION MODEL ARCHITECTURE

Antonín KAVIÈKA

The paper describes the original architecture of simulation model, which is based on the concept of cooperating autonomous agents (it is called ABAsim – Agent-based Architecture of simulation model). Each agent represents the independent control module that is responsible for management of determined model section (its state space). In case of the problem occurrence, which is not able to solved by the mentioned agent itself, there is utilised the cooperation with other agents.

The agents are composed of specialised internal components, which are focused on the execution of the individual agent functions. The agent behaviour can be very flexibly modified by means of the exchange of its required components i.e. the entire simulation model provides high degree of flexibility.

There are four groups of internal agent components. Group of *control* and *decision components* consists of a single component type called *manager*. Each agent has one and only one manager. This component of an agent assumes the capability of the objective recognition, capability of communication with other agents and other components inside the same agent and the decision making capability. Manager models the intelligence of an agent.

Access to the information on the state of a system is provided by the group of *informers* which consists of components of two types. *Query* produces immediately an information (solicited by a manager), while *monitor*, after its initialization, operates independently and repeatedly investigates the chosen aspects of the state of the system and provides manager with necessary information.

Group of *solution makers* supports decisions of a manager by suggestions how to solve a problem. *Adviser* is a passive component which, on request of a manager, immediately proposes a solution(s). Adviser uses usually an optimization algorithm, consultation with an expert system or consultation with the plan of solutions. To create a plan of solutions means to anticipate the possible occurence of problems and, in a suitable time horizont, to suggest their solution (e.g. time schedule for allocating the servers). Component called *planner* is able to create such solution plans and to update them in chosen time points or in specified situations even without manager authorization.

The last group of components are *executors*. They are executing the decisions of managers concerning the state of the system. No other components can change the state of the system. Component called *action* executes immediate one stroke change of the system (e.g. voltage increasing, switch change), while *process*, after its activation, autonomously, in specified time points, changes the values of state variables (e.g. switches regularly signal lights on crossings). Executors, informers and solution makers are called *effectors*.

From the functional and implementation point of view effectors can be divided into two following groups:

- *time-dependent effectors*, which consume time, it means that after being triggered they "live" independently and during their life they have capability to accept and to send messages (processes, monitors, planners) and

- *one stroke effectors* (not consuming time) which are executed immediately in the moment of their triggering (actions, queries, advisers).

The most important result for the user, which the ABAsim methodology brings him in the development of a model, is its openness in the stage of current use of a simulation model. User can have in a database a number of alternatives for each manager and effector and may choose among them in the setting of the scenario of a simulation run. Thus a user can modify executive properties of an agent (by a choice of its executors), decision making properties of an agent (choosing alternative set of informers and solution makers), control strategy (choosing alternative set of managers) an even can modify modelling of a section of the real world (by choosing alternative agents).

Because human operator (user) is also one of the agents, a mode of cooperation of the user with other agents can be chosen and modified. This concerns mainly the way of problem solution, which can be modified in a broad scope of possibilities, from fully automatic (machine made) up to a "hand made" (user).

The described ABAsim concept provides set of advantages; let us mention at least the most important ones:

- close similarity of the structure of a model with modelled system in all stages of its development,
- high degree of the system modularity and the possibility to use individual modules in different models,
- possibility to integrate human operator into the system (as one of agents),
- support for creation of universal models rather than construction of one-purpose simulation models,
- possibility to modify configuration of the model and simulation scenarios without intervention into the program code,
- support to connection the models into a network, which enables to construct models of networks and also a network of models.

The ABAsim architecture was applied with encouraging results within the frame of development of complex railway marshalling yard simulation model. The mentioned model was utilised in the number of real projects (for different railway companies), which were focused on the simulation modelling of big railway stations located in Western Europe and China. The experience from those projects entirely confirmed the above-mentioned advantages of ABAsim architecture.

Zusammenfassung

DIE ARCHITEKTUR DES SIMULATIONSMODELLS GEGRÜNDET AUF DEM PARADIGMA DER AUTONOMEN AGENTEN

Antonín KAVIÈKA

Der Artikel beschreibt eine Originalarchitektur des Simulationsmodells. Diese Architektur wird auf dem Konzept der kooperativen Agenten gegründet. Jeder Agent repräsentiert einen autonomen Steuermodul, der verantwortlich für die “Verwaltung“ des konkreten Modelteils (Zustandsraums) ist. Wenn ein Problem vorkommt, den der Agent nicht kann lösen, kooperiert er mit anderen Agenten.

Die Agenten bestehen aus den speziellen Komponente, die einzelne Funktionen des Agenten ausführen. Die Wirkung des Agenten ist möglich sehr flexibel durch den Komponentewechsel zu modifizieren, d.h. das ganze Modell weist die hohe Flexibilität auf. Das besagte Konzept des Simulationsmodells bietet viele Vorteile um z.B.:

- die Modellstruktur sieht sich nach der Struktur des modellierten Systems auf,
- die Modellmodularität ermöglicht die Modulbenutzung in den anderen Simulationsmodelle,
- eine kunstlose Integration des Anwenders in dem Model (einer von den Agenten),

- es wird der Ausbau vielmehr des generellen Simulationsmodells als des Einzweckmodells unterstützt,
- der Modellkonfigurationswechsel ist möglich ohne den Programmcodewechsel – es wird mit Hilfe der verschiedenen Editoren realisiert,
- es wird die Modellausbreitung unterstützt d.h. die Modelle werden leicht in der Netze verbunden.

Die Agentarchitektur des Simulationsmodells wurde mit einem Erfolg appliziert – es wurde z.B. ein komplexer Simulationsmodell der Zugbildungsanlage (des Rangierbahnhofs) ausgebaut und in viele realen Projekten in Europa und China ausgenützt.