HYBRIDNÍ ARCHITEKTURA PRO WEBOVÉ SIMULÁTORY

Tomáš Kulhánek, Marek Mateják, Jan Šilar, Pavol Privitzer, Martin Tribula, Filip Ježek, Jiří Kofránek

Anotace

Příspěvek se zabývá simulátory běžícími na serveru a poskytujícími hrubá data, která teprve klient (např. webový prohlížeč nebo specializovaná aplikace) vizualizuje. Takováto hybridní architektura nabízí nasazení dlouhotrvajících simulací a výpočetně náročných operací na výkonné servery přičemž sledování výsledků simulace lze prohlížet jak na běžných počítačích tak i na mobilních zařízeních typu tablet. Architektura byla zkoušena s komplexním fyziologickým modelem Hummod a simulací acidobazických poruch a jejich dlouhodobější trendy.

1. Úvod

Na konferencích MEDSOFT roku 2010, 2011 a 2012 Kofránek a spol. shrnuli vývoj technologií pro tvorbu simulátorů se zvláštním zřetelem na výuku a výzkum v oblasti medicíny, přičemž z vývoje tzv. nativních simulátorů se postupně přeorientovali na vývoj tzv. webových simulátorů běžících v internetovém prohlížeči[1]. Simulátory využívají technologii Adobe Flash nově i technologii Microsoft Silverlight, kvůli lepší integraci s modely a výkonu výpočetně náročných komplexnějších modelů[2, 3].

2. Nativní simulátory

Nativní simulátor je aplikace, která je přeložena do jazyka procesoru cílového počítače a operačního systému a spouští se přímo, tj. nativně. Distribuce nativních aplikací dnes probíhá např. formou instalačního balíčku distribuovaného přes klasická média (CD-ROM, DVD-ROM). Mezi nativní aplikace se řadí i ty, které lze stáhnout a nainstalovat z internetu ať už stažením instalačního balíčku, či technologiemi usnadňujícími spuštění aplikace bez instalace (např. Java Web-Start, enbo Microsoft ClickOnce). Nativní simulátor obvykle dál nevyžaduje připojení k internetu. Mezi takové simulátory lze zařadit např. komplexní simulátor ledvin vyvinutý Kofránkem a Tribulou v prostředí Control Web[4] nebo výukovou aplikaci regulací vyvinutou Wünschem a spol. [5]. Obě jsou součástí atlasu patologické fyziologie (*www.physiome.cz/atlas*) [6]. Podobně jsou koncipovány simulátory pocházející z jiných univerzit, např. simulátor ECGSIM vyvinutý v Holandsku na Radboud University in Nijmegen (*www.ecgsim.org*) [7, 8], který umožňuje simulovat šíření elektrického potenciálu v komorách a síních srdce a měnit šíření vzruchu myokardem, což je typické pro některé patologie. Nebo program pro výpočet a zobrazení pH krve a koncentrace krevních plynů - Oxygen Status Algorithm [9] nyní ve verzi 1.0 pro Windows (*http://www.siggaard-andersen.dk*).

Výhodou nativních simulátorů je přímý přístup k lokálním zdrojům a vysoký výkon limitovaný pouze možnostmi lokálního počítače. Nevýhodou je nutnost jejich instalace a údržby na příslušné platformě, což vyžaduje jisté znalosti od uživatele a také větší práva např. na spouštění a zápis na disk.

3. Webové simulátory

S rozvojem technologií umožňující vývoj tzv. Rich Internet Application (RIA) [10] se webové aplikace stále více podobají desktopovým aplikacím, přičemž webové aplikace mají některé výhody vhodné pro výukové a vědecké simulátory. Byrne a spol.[11] ve své práci z roku 2010 mapují webové simulátory (nikoliv nativní simulátory) z širší oblasti a rozlišují u webových simulátorů tři následující typy architektur.

3.1 Vzdálená simulace a vizualizace

Simulace včetně renderování výsledku je prováděno na serveru a internetový prohlížeč pouze zobrazí výslednou grafiku, či text.[11]

Mezi technologie umožňující běh složitější aplikace na serveru a renderování výsledku v podobě obrázků a HTML pro klienta patří technologie CGI[12], či přímé moduly pro skriptovací jazyky PHP, Perl a propojení webového serveru s vývojovými platformami ASP.NET, J2EE, Ruby on Rails a další.

Vzdálená simulace a vizualizace vyžaduje stálé připojení uživatele k internetu a je náročné na rychlost a zpoždění komunikace mezi serverem a klientem. Tato vlastnost byla limitující zvlášť v dřívějších dobách, kdy vysokorychlostní internet nebyl široce dostupný.

3.2 lokální simulace a vizualizace

Simulační jádro a vizualizační komponenty jsou staženy internetovým prohlížečem do lokálního počítače uživatele a simulace včetně vizualizace je dále prováděna už bez interakce se serverem v rámci internetového prohlížeče. [11] Technologie umožňující běh aplikací v prohlížeči reagují na to, že samotný jazyk dokumentů HTML a protokol HTTP nestačí na požadavky kladené na webové aplikace a tudíž se vyvinula řada technologií, které podporují přímo prohlížeče (Javascript), případně se dají doinstalovat formou pluginu (MS Silverlight, Adobe Flash, ...). Uvolněním nové verze HTML (verze 5) a její podpora ve většině nových verzí internetových prohlížečů se vývoj a dostupnost RIA ještě více rozšířila.

Lokální simulace a vizualizace není náchylná na rychlost a zpoždění serveru a při připojení více klientů k jednomu serveru není server zahlcen paralelním simulováním a renderováním, proto tento koncept je již od roku 2005 sledován při vývoji simulátorů pro atlas patologické fyziologie *www.physiome.cz/atlas*[6, 13, 14]. Od vývoje simulátorů běžících na klientovi v technologii Adobe Flash se dnes preferuje např. technologie MS Silverlight, která nabízí vyšší výkon a lepší integraci s modelem a vývojovými a grafickými prostředky[2].

3.3 hybridní simulace a vizualizace

Na serveru probíhá simulace a klient obdržená data vizualizuje víceméně v reálném čase[11]. Technologie použitá pro tento typ hybridní architektury webových simulátorů kombinuje technologie zmíněné v předchozích dvou odstavcích pro vzdálenou a pro lokální simulaci/vizualizaci. Simulační a vizualizační vrstva řeší formát a výměnu dat přes internet.

Hybridní architektura dovolí pouštět dlouhodobé simulace či výpočty na serveru a jejich sledování, či ovlivňování na klientovi. Vizualizace může být prováděna i na méně výkonných zařízeních např. typu tablet. Níže popsaný prototyp simulátoru komplexního modelu Hummod používá tento typ architektury.

4 simulátor komplexního modelu fyziologie

Hester a spol. publikovali model Hummod, rozsáhlý fyziologický model zkonstruovaný z empirických dat z recenzované vědecké literatury a popsaný v jazyce XML, který může být později modifikován uživatelem pro zkoumání a testování nových přístupů[15]. Tento popis v jazyce XML je příliš komplexní pro porozumění a Mateják a spol. publikovali jeho implementaci v akauzálním jazyce Modelica a navíc ho upravili a rozšířili zejména v oblasti fyziologie vnitřního prostředí[16].

Jazyk Modelica je v současnosti standardem[17] a existuje celá řada nástrojů od různých poskytovatelů, které ho implementují a umožňují editaci a základní simulaci, např. open source OPENMODELICA *https://www.openmodelica.org*[18] vyvíjený konsorciem Open Modelica, DYMOLA vyvíjená firmou Dassault Systemes *http://www.dymola.com*, SystemModeler vyvíjený firmou Wolfram *http://www.wolfram.com/system-modeler/* a další. Tyto nástroje dovolují provádět základní simulaci a vizualizaci výsledků v grafech. Nicméně pro tvorbu složitějších simulátorů je nutné použít nějakou formu exportu a integrace s nástroji pro vývoj softwaru.

4.1 integrace se simulátorem

Existuje několik přístupů jak model začlenit do aplikace a vytvořit tak funkční simulátor. Modelovací nástroje nabízejí dva přístupy. První dovoluje exportovat konkrétní model do samostatné aplikace spustitelné na příkazové řádce. Tento typ integrace se obvykle popisuje jako integrační vzor File-Transfer[19]. Je to jednoduchý typ integrace pro vybudování rozsáhlejšího systému z nezávislých aplikací, pokud umístění a formát dat jsou známé, nicméně neumožňuje ovládat simulátor v reálném čase.

Druhý přístup dovoluje exportovat model do knihovny přístupné přes aplikační rozhraní, které se dá volat ze zvolené platformy. Dymola dovoluje ovládat exportované modely jako DLL knihovny pomocí svého vlastního API v jazyce C. Nástroj OpenModelica a jeho překladač a řešič v technologii .NET dovoluje export modelu do knihovny ovládané přes rozhraní dostupné pro jiné aplikace vyvíjené v technologii .NET.

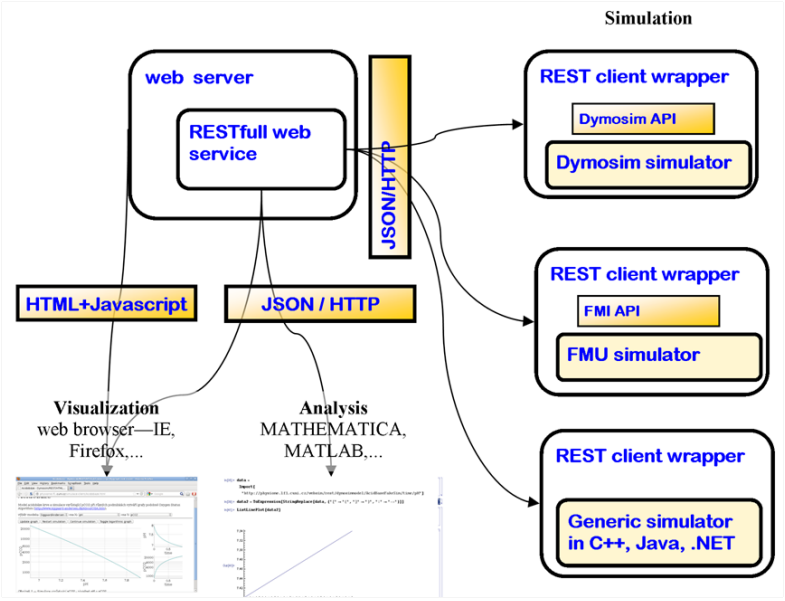
Poskytovatelé modelovacích nástrojů včetně Dymoly a jejich partneři vytvořili a udržují standard Functional Mockup Interface (FMI), který dovoluje výměnu modelů a simulátorů mezi různými spotřebiteli a dodavateli standardní formou přes tzv. FMI API a použít různé na Modelice založené nástroje[20].

Pro vytvoření rozsáhlejších systémů z relativně nezávislých aplikací se ještě často používá integrační vzor založený na zasílání zpráv tzv. Messaging Pattern[19]. Některé nástroje jazyka Modelica nabízejí integraci založenou na zasílání zpráv a standardech Dynamic Data Exchange (DDE) a průmyslový

OLE for Process Control (nebo též Open Platform Communication OPC (www.opcfoundation.org). Nicméně pro potřeby webového simulátoru jsme zvolili alternativu komunikace přes protokol HTTP s tzv. REST webovou službou.

Pro potřebu ověření hybridní architektury webového simulátoru jsme vyvinuli webovou službu (web service), které skloubuje architektonický styl REST[21], ve kterém data a stavy simulátoru jsou přístupné a ovladatelné přes web pomocí protokolu HTTP, formátu dat Javascript Object Notation (JSON) a omezeného počtu operací create, read, update, delete (CRUD). Tato webová služba na základě požadavků klienta vrací data simulace, která má buď již uložena v databázi, případně spustí modul hrubého simulátoru (exportovaný model pomocí výše zmíněných přístupů) spustí. Klient této webové služby je vizualizační aplikace, která umí přes protokol HTTP posílat a zpracovávat data ve formátu JSON. V našem případě jsme vytvořili stránku v HTML verzi 5 používající Javascript a některé open source knihovny JQuery (www.jquery.com) pro asynchronní komunikaci s REST webovou službou, DyGraph (www.dygraph.com) pro vykreslování grafů ze získaných dat, a další. Klientem této webové služby je i simulátor exportovaný z Modelicy a přes standardní (FMI) nebo proprietární (Dymola - Dymosim) API je přidána vrstva která umí s REST webovou službou pomocí protokolu HTTP komunikovat.

Webová služba poskytuje repozitář modelů, které lze simulovat. Dále poskytuje data simulací vztahující se ke konkrétnímu modelu a poskytuje



Obrázek 1 — Architektura REST webové služby a simulací prováděných na serveru a vizualizace a zpracování dat prováděné na klientovi

je ve formě pole s proměnnými seřazena podle času simulace a podle kritérií požadována uživateli, tj. vybrané proměnné a pořadí v poli. Webová služba navíc poskytuje repozitář vizualizací, které se předkládají webovému browseru ve formě HTML stránek s Javascriptem. Vizualizace jsou popsané ve vlastním doménově specifickém jazyce, který je stále ve vývoji.

V případě jednoduchých modelů webovému serveru stačí ukládat tato data dočasně do paměti a případně si ukládat některé výsledky simulací do souboru. V případě simulace komplexního modelu Hummod, který simuloval průběhy všech známých veličin po dobu jednoho simulovaného měsíce s frekvencí jeden záznam všech hodnot za několik málo sekund, data této simulace se nevešla do interní paměti (několik GB dat). Webová služba byla rozšířena o správu a ukládání simulačních dat do databáze - používáme Entity Framework k přístupu k datům v databázi MS SQL.

5 Diskuze

Navržená hybridní architektura si samozřejmě nedělá ambice nahradit dosavadní technologie výukových aplikací, které simulují a vizualizují na lokální straně. Hybridní architektura ale může rozšířit portfolio simulátorů, které jsou dnes součástí atlasu patologické fyziologie o aplikace s dlouhodobými simulacemi rozsáhlých modelů a rozsáhlými výpočty, které jsou potřeba např. v případě identifikace fyziologických systémů. Atlas může být rozšířen o repozitář dat, která zatím v atlasu patologické fyziologie nejsou, tj. zdrojových kódů modelů, jejich popis, definice vizualizací a simulací spustitelných na dálku, atd. Volné spojení simulačních modulů s webovou službou nevyžaduje její spuštění na tomtéž serveru, ale může být delegováno na jiné stroje připojené k internetu, např. vysoce výkonnou kapacitu gridu a cloudu poskytovaného sdružením CESNET - aktivitou METACENTRUM (*meta.cesnet.cz*), případně evropskou iniciativou EGI, propojující kapacity pro vědecké účely na evropské úrovni (*www.egi.eu*). Pro tyto potřeby bude nutné aplikovat na server robustní řešení zabraňující zneužití kapacit této služby k jiným než simulačním účelům např. formou přihlašování při použití potencionálně zatěžujících dlouhodobých simulací.

Poděkování

Tato práce vznikla za podpory grantu MPO FR-TI3/869 a Fondu rozvoje sdružení CESNET z.s.p.o.

Literatura

[1.] J. Kofránek. Webové simulátory. sborník příspěvků Medsoft 2010, pages 81–95, 2010.

[2.] J. Kofránek, M. Mateják, F. Ježek, P. Privitzer, and J. Šilar. Výukový webový simulátor krevního oběhu. sborník příspěvků Medsoft 2011.

[3.] J. Kofránek. Integrované modely fyziologických systémů - historie, současnost, perspektivy. sborník příspěvků Medsoft 2012.

[4.] J. Kofránek and M. Tribula. Control web pro multimediální interaktivní ledvinu. sborník příspěvků Medsoft 2007.

[5.] Zdeněk Wünsch, Marcel Matúš, Tomáš Kripner, and Jiří Kofránek. Modely regulace ve fyziologickém praktiku. sborník příspěvků Medsoft 2007.

[6.] M. Andrlík, J. Kofránek, S. Matoušek, P. Stodulka, Z. Wünsch, T. Kripner, and J. Hlaváček. Internetový atlas výukových multimediálních modelů pro vybrané kapitoly normální a patologické fyziologie člověka. ukázka předběžných výsledků. sborník příspěvků Medsoft 2006.

[7.] Allan Van Oosterom and TF Oostendorp. Ecgsim: an interactive tool for studying the genesis of qrst waveforms. Heart, 90(2):165–168, 2004.

[8.] Peter M van Dam, Thom F Oostendorp, and Adriaan van Oosterom. Interactive simulation of the activation sequence: replacing effect by cause. pages 657–660, 2011.

[9.] O Siggaard-Andersen and M. Siggaard-Andersen. The oxygen status algorithm: a computer program for calculating and displaying ph and blood gas data. Scand J Clin Lab Invest, pages 29–45, 1990.

[10.] Piero Fraternali, Gustavo Rossi, and Fernando Sánchez-Figueroa. Rich internet applications. Internet Computing, IEEE, 14(3):9–12, 2010.

[11.] J. Byrne, C. Heavey, and P. J. Byrne. A review of web-based simulation and supporting tools. Simulation Modelling Practice and Theory, 18(3):253–276, 2010.

[12.] Shishir Gundavaram. CGI Programming on the World Wide Web. O'Reilly, 1996.

[13.] J. Kofránek, M. Andrlík, T. Kripner, and P. Stodulka. From Art to Industry: Development of Biomedical Simulators. The IPSI BgD Transactions on Advanced Research, 1(2):62–67, 2005.

[14.] J. Kofránek, M. Andrlík, T. Kripner, J. Hlaváček, and P. Stodulka. E-learning s výukovými simulačními modely v prostředí macromedia breeze. sborník příspěvků Medsoft 2005.

[15.] Robert Hester, Alison Brown, Leland Husband, Radu Iliescu, William Andrew Pruett, Richard L Summers, and Thomas Coleman. Hummod: A modeling environment for the simulation of integrative human physiology. Frontiers in Physiology, 2(12), 2011.

[16.] M. Mateják and J. Kofránek. Hummod - golem edition - rozsáhlý model fyziologických systémů. sborník příspěvků Medsoft 2011.

[17.] P. Fritzson and Brunus P. Modelica – a general object-oriented language for continuous and discrete-event system modeling and simulation. Simulation Symposium, 2002. Proceedings. 35th Annual, pages 365–380.

[18.] Peter Fritzson, Peter Aronsson, Adrian Pop, Hakan Lundvall, Kaj Nystrom, Levon Saldamli, David Broman, and Anders Sandholm. Openmodelica-a free open-source environment for system modeling, simulation, and teaching. In Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE, pages 1588–1595. IEEE, 2006.

[19.] Gregor Hohpe and Bobby Woolf. Enterprise integration patterns: Designing, building, and deploying messaging solutions. Addison-Wesley Professional, 2004.

[20.] Torsten Blochwitz, M Otter, M Arnold, C Bausch, C Clauß, H Elmqvist, A Junghanns, J Mauss, M Monteiro, T Neidhold, et al. The functional mockup interface for tool independent exchange of simulation models. In Modelica'2011 Conference, March, pages 20–22, 2011.

[21.] Roy Thomas Fielding. Chapter 5: Representational state transfer (rest). Architectural Styles and the Design of Network-based Software Architectures, Dissertation, 2000.

Kontakt:

Tomáš Kulhánek
Oddělení biokybernetiky a počítačové podpory výuky
ÚPF 1. LF UK, Praha
U nemocnice 5,
128 53 Praha 2

Marek Mateják
Ústav patologické fyziologie 1.LFUK

Jan Šilar
Ústav patologické fyziologie 1.LFUK

Pavol Privitzer
Ústav patologické fyziologie 1.LFUK

Martin Tribula
Ústav patologické fyziologie 1.LFUK

Filip Ježek
FEL ČVUT

Jiří Kofránek
Ústav patologické fyziologie 1.LFUK