MOKSLAS – LIETUVOS ATEITIS
SCIENCE – FUTURE OF LITHUANIA

ISSN 2029-2341 print / ISSN 2029-2252 online
2013 5(2): 92–95 doi:10.3846/mla.2013.16

Sistemų inžinerija, kompiuterių technologija T 120
Systems Engineering, Computer Technology T 120

Elektronika ir elektrotechnika
Electronics and Electrical Engineering

REST ARCHITEKTŪRINIO STILIAUS PALYGINIMAS SU SOAP, ĮGYVENDINANT ŠIUOLAIKINES INTERNETO PASLAUGAS

Kęstutis Stankevičius
Vilniaus Gedimino technikos universitetas
El. paštas: kestutis.stankevicius@vgtu.lt

Santrauka. Interneto paslaugų įgyvendinimas, siekiant perkelti programų funkcionalumą į interneto paslaugų sąsają, yra vienas aktualiausių informacinės sistemos kūrimo etapų. Interneto paslaugos tapo neatsiejama programinės įrangos dalimi, bet blogi paslauginių architektūrų įgyvendinimo rezultatai verčia susimąstyti. Informacinių sistemų pardavėjams palankios specifikacijos verčia įmones ieškoti alternatyvų, patenkinančių jų nustatytus poreikius, t. y. pigesnių, efektyvesnių ir galbūt greičiau įgyvendinamų. Straipsnyje bandoma palyginti standartinį būdą teikti interneto paslaugas su kitu architektūriniu interneto paslaugų įgyvendinimo stiliumi. Pastarasis vis plačiau taikomas jau nuo 2008 m.

Reikšminiai žodžiai: interneto paslaugos, SOAP, paslauginė architektūra, REST, architektūrinis stilius.

Įvadas

Interneto paslaugos (angl. *Web services*) paskutiniu metu yra labai aktuali tema programinės įrangos kūrimo srityje. Interneto paslaugų konstrukcija leidžia įvairių aplinkų sistemoms sąveikauti per tinklą, t. y. interneto paslaugos leidžia skirtingoms programoms bendrauti tarpusavyje, net jeigu jos parašytos skirtingomis programavimo kalbomis ar veikia ant skirtingų platformų. Taikomųjų programų pardavėjus ypač domina interneto paslaugų naudojimas siekiant perkelti egzistuojantį programų funkcionalumą į interneto paslaugų sąsają, kur programos gali būti panaudotos tiesiogiai arba integruotos į kitą programą, taip sukuriant visiškai naują funkcionalumą. Interneto paslaugų potencialas ir paslauginės architektūros paskatino atlikti daug tyrimų, bandymų, taip pat sukėlė diskusijas ir paskatino įvairių standartų bei specifikacijų atsiradimą.

Taigi interneto paslaugos tapo neatsiejama programinės įrangos dalimi, bet pagal *Information Week* atliktą interneto apklausą, kurioje dalyvavo 287 informacinių technologijų profesionalai, net 32 % pasakė, kad jų SOA (angl. *Service Oriented Architecture*) projektai nepateisino jų lūkesčių, o likusieji 58 % pasakė, kad SOA padarė jų IT ūkį daug kartų sudėtingesnį, o iš tų 58 % net 30 % pridūrė, kad SOA kainavo daugiau nei tikėtasi ir skeptiškai žiūrėjo į SOA investicijų grąžą (angl. *Return of Investment*). Dar vienas *Information Week* atliktas tyrimas, kurį sudarė 106 organizacijos, įgyvendinusios SOA, atskleidė, kad tik 37 % iš tų organizacijų pajautė bent kokią investicijų grąžą (Smith 2008).

Interneto paslaugų įgyvendinimas SOA būdu yra brangus ir ganėtinai kompleksiškas, be to, atima nemažai laiko. Tiesą sakant, daugelis teikia pirmenybę SOAP (angl. *Service Oriented Architecture Protocol*) metodui įgyvendinant interneto paslaugas, nes vyrauja nuomonė, kad tai vienintelis būdas įgyvendinti interneto paslaugas. SOAP tinkamesnis didelėms organizacijoms, o mažiau sudėtingoms interneto paslaugoms įgyvendinti vis plačiau naudojamas REST (angl. *Representational State Transfer*) architektūros stilius. Nors REST ir neturi patvirtintų specifikacijų kaip SOAP, šie du interneto paslaugų įgyvendinimo metodai sukėlė karštus debatus programinės įrangos kūrimo bendruomenėje: kuris metodas geresnis? Šiame straipsnyje apibendrinama SOAP ir REST realizacija, akcentuojama, kodėl REST technologiją naudoti yra geriau.

Šio straipsnio tikslas – išanalizuoti SOAP ir REST taikomas metodikas interneto paslaugoms įgyvendinti ir parodyti, kodėl REST yra geriau, įgyvendinant šiuolaikines interneto paslaugas.

Tikslui pasiekti suformuluoti uždaviniai:

1. Apžvelgti šiuo metu plačiausiai interneto paslaugoms įgyvendinti taikomus SOAP ir REST metodus.
2. Nustatyti SOAP ir REST technologijų pranašumus bei trūkumus.
3. Palyginti SOAP ir REST interneto paslaugų įgyvendinimo principus.

Interneto paslaugų įgyvendinimas

Interneto paslaugų specifikacijos, esančios *World Wide Web* konsorciume (W3C), siūlo naudoti karkasą, pagrįstą SOAP paslaugų bendravimu ir WSDL (angl. *Web Service Description Languag*e) paslaugų aprašymu. WSDL yra XML (angl. *eXtensible Markup Language*) dokumentas, kuris apibūdina interneto paslaugų sąsajas kartu su susiejimo informacija į tinklo protokolą, tokį kaip HTTP (angl. *Hyper Text Transfer Protocol*). SOAP yra XML pagrįstas pranešimų protokolas, kuris veikia virš esamo tinklo protokolo, tokio kaip HTTP. Interneto paslaugos klientas per SOAP pirmiausia įgyja interneto paslaugos WSDL failą. WSDL numato klientui baigties kriterijų, galimų operacijų sąrašą ir pranešimų formatus kiekvienai operacijai. Tada klientas gali sukurti SOAP žinutę ir kreiptis į paslaugą, kad iškviestų vieną iš galimų operacijų.

Alternatyvus būdas kurti ir prieiti prie interneto paslaugų nereikalauja jokio papildomo karkaso, be jau sukurtų ir naudojamų šiandien, t. y. HTTP ir URI (angl. *Uniform Resource Identifier*), čia URI yra bet kokio ištekliaus identifikatorius pagal vietą arba pavadinimą, o kad nesupainiotume su URL (angl. *Uniform Resource Locator*), tai URL yra URI specializacija, kuri apibrėžia konkretų tinklo šaltinį ir naudojamą protokolą, t. y. nurodo, kaip galima gauti URI.

REST (angl. *Representational State Transfer*) yra alternatyvus architektūrinis stilius, skirtas interneto paslaugoms kurti gaunant ir manipuliuojant ištekliais su nustatytomis standartinėmis operacijomis, tokiomis kaip GET, POST, PUT ir DELETE, bet jis, kitaip nei SOAP, per žinutes neiškviečia potencialiai didelio operacijų rinkinio. REST interneto paslaugos klientas tiesiog prisijungia prie interneto paslaugos per URI ir interneto paslauga grąžina reikiamus išteklius. Grąžinti ištekliai gali būti XML dokumentas, kuriame yra visas operacijų sąrašas, naudojamų operacijų schema ir URI prieiga prie kiekvienos operacijos. Klientas gali iškviesti vieną iš galimų operacijų iš sąrašo ir taip tęsti bendravimą su interneto paslauga.

SOAP pranašumai ir trūkumai

SOAP šalininkai teigia, kad SOAP suteikia didelį lankstumą projektuojant interneto paslaugas. SOAP interneto paslauga gali turėti gausybę iškviečiamų operacijų, o WSDL šablonas teikia informaciją, kurios reikia klientui, norinčiam naudoti tą SOAP interneto paslaugą. Tai suteikia galimybę iškviesti vieną iš daugelio viešai prieinamų operacijų – programuotojams tai gerai pažįstamas metodas, pagrįstas API (angl. *Application Programming Interface*) būdu, realizuojant interneto paslaugas.

SOAP turi nemažai išleistų standartų ir specifikacijų, kuriais remiasi įvairios korporacijos. Tai leidžia įmonėms kurti priemones, kurios padėtų projektuoti SOAP interneto paslaugas. Iš tikrųjų šios priemonės atlieka beveik visą kompleksišką darbą, taip leidžiant kūrėjams sutelkti dėmesį į SOAP interneto paslaugos funkcionalumą. SOAP žinutės turi apsaugos mechanizmą, tad duomenys garantuotai nebus matomi tiems, kurie neturi teisių jų matyti, taip sukuriant saugų ryšį tarp visų galimų jungčių, kurias naudoja SOAP interneto paslauga.

SOAP šalininkai teigia, kad reikia taikyti jų būdą, nes jis labai lankstus projektuojant interneto paslaugas, bet taip pat tai yra viena priežasčių, kodėl nereikia naudoti SOAP. Dėl to, kad kiekviena SOAP interneto paslauga turi savo rinkinį operacijų, reikia, kad klientas suprastų kiekvieną iš visų naudojamų interneto paslaugų operacijų rinkinių. Tai reiškia, kad klientas, bendraudamas su konkrečia kiekvienos paslaugos operacija, turi išmanyti jos duomenų struktūrą.

SOAP interneto paslaugos naudoja tik HTTP POST operaciją, o visi kiti duomenys apie atliekamą operaciją yra sudėti į SOAP žinutę (Prescod 2002). Sistemos administratorius negali taip lengvai atlikti savo darbo, nes visa informacija apie operaciją paslėpta SOAP žinutėje. Tokiomis sąlygomis gali atsirasti potencialiai naujų saugumo skylių, nes SOAP atveju sistemos administratoriui reikėtų peržvelgti kiekvieną žinutę atskirai siekiant nustatyti, kaip operacija atliekama.

Turėti saugias žinutes, siekiant apsaugoti duomenis, jei tie duomenys patenka už saugių perdavimo kanalų ar ryšių, yra labai gera idėja. SOAP tai realizuoja naudodama karkasą, apibrėžtą *WS-Security* specifikacijoje. Ši specifikacija naudoja XML šifravimo (angl. XML *Encryption*) ir XML parašo (angl. XML *Signature*) specifikacijas apsaugant XML dokumentus.

SOAP pranašumai:
- SOAP žinutės;
- SOAP žinučių perdavimo mechanizmas.

SOAP trūkumai:
- sudėtingumas;
- besikeičiančios specifikacijos;
- pardavėjų poreikių skatinamos specifikacijos;
- reikia sudėtingų priemonių ir karkasų sudėtingumui palengvinti.

REST pranašumai ir trūkumai

REST reikalauja galvoti kitaip, o ne įprastais metodais, tokiais kaip parametrų perdavimas arba SOAP atveju sudėtingų žinučių pasitelkimas duomenims perduoti. REST

sutelkia dėmesį į išteklius, o ne į patį API. Bet iš tikrųjų REST nėra toks jau neįprastas ir nepažįstamas programuotojams. Imkime kaip pavyzdį SQL (angl. *Structured Query Language*) reliacinę duomenų bazę, kur lentelės yra ištekliai, o jau žinomi SELECT, UPDATE, INSERT ir DELETE sakiniai – operacijos. Tada tai visai pažįstamas metodas programuotojams, kurie taiko HTTP operacijas GET, POST, PUT ir DELETE norėdami pasiekti REST interneto paslaugas, gauti ir manipuliuoti XML dokumentais – labai panašiai kaip su SQL duomenų baze. Interneto paslaugų projektavimas ir įgyvendinimas naudojant REST yra didelio lankstumo garantas. Tik su mažu operacijų rinkiniu galima pasiekti norimą skaičių išteklių, kurie prieinami per URI (Prescod 2002).

Galima teigti, kad kurti programas naudojant REST interneto paslaugas yra daug lengviau negu SOAP. REST nereikalauja jokių specializuotų priemonių, išskyrus jau turimas, kurios leidžia naudotis tinklo ištekliais per HTTP ir dirbti su XML dokumentais. Priemonės, naudojamos REST interneto paslaugoms įgyvendinti, yra lengvai prieinamos, pavyzdžiui, naudojant įprastą interneto naršyklę galima atlikti derinimo veiksmus (angl. *Debugging*) (Kay 2007). Įmonių susidomėjimas REST interneto paslaugomis jau atsiranda kaip ir susidomėjimas Java API, kuris turi XML interneto paslaugas, žinomas kaip JAX-WS, kur galima rasti programos kodo pavyzdžių, kaip įgyvendinti REST interneto paslaugas (Hinchcliffe 2005). Vis dėlto REST metodui trūksta palaikymo iš komercinių kompanijų (Tyagi 2006).

Jei norima saugiai autentifikuoti klientą ir leisti jam naudotis arba manipuliuoti interneto paslaugų ištekliais, REST interneto paslaugas galima apsaugoti perdavimo sluoksnyje įprastu HTTP metodu. Kadangi REST interneto paslaugos praktiškai keičiasi XML dokumentais, tai XML šifravimo ir XML parašo specifikacijos taip pat gali būti naudojamos ir pritaikomos siunčiamiems duomenims apsaugoti.

REST interneto paslauga, naudodama HTTP protokolą, yra palanki sistemos administratoriams, nes jie gali matyti, kokios paslaugos naudojamos ir kas jas naudoja tiesiogiai. Be to, HTTP operacijos gali būti lengvai apribotos tam tikriems URI norint, kad, pavyzdžiui, paslauga galėtų būti tik peržvelgta, o ne manipuliuojama. Tokiu būdu interneto paslaugos projektuotojas ir sistemos administratorius yra kaip viena komanda. Sistemos administratorius gali dirbti tiesiogiai be jokių papildomų priemonių.

REST pranašumai:

- paprastumas;
- patikrintas architektūrinis stilius (internetas);
- mažiau kūrimo etapų, metodinės medžiagos ir vykdymo variklių.

REST trūkumai:

- nėra nustatytų specifikacijų;
- nėra priimto atvaizdo šablono kaip WSDL.

REST ir SOAP palyginimas

SOAP žinutės gali būti apsaugotos ir gali naudoti perdavimo lygio saugumą, numatytą HTTPS (angl. *Hyper Text Transport Protocol Secured*). REST gali naudoti tik perdavimo lygio saugumą, kurį teikia HTTPS (Comerford 2010). Naudojant perdavimo lygio saugumą užtikrinama duomenų apsauga tik tarp kliento ir interneto paslaugos, o tai reiškia, kad duomenys gali būti neapsaugoti, kai interneto paslauga bendrauja su kitomis interneto paslaugomis, programomis ar verslo partneriais, nes nėra jokios garantijos, kad perduodama HTTPS būdu.

SOAP interneto paslaugos gali būti įgyvendintos naudojant paskelbtas specifikacijas. Būtent dėl šių specifikacijų SOAP interneto paslaugų įgyvendinimas yra kompleksiškesnis, nei naudojant REST interneto paslaugų architektūrą. Tiesa, egzistuoja daug įvairių priemonių, kurios padeda kurti SOAP interneto paslaugas, laikantis nurodytų specifikacijų, tačiau tos priemonės ir yra reikalingos dėl papildomo kompleksiškumo. Programinės įrangos kūrėjui įgyvendinti SOAP interneto paslaugą be jokių papildomų priemonių būtų labai sudėtinga: įsivaizduokite, jeigu jam reikėtų sukurti WSDL ar SOAP žinutę naudojant įprastą tekstinį redaktorių.

Nors įdėta nemažai pastangų plečiant SOAP galimybes, ypač kad SOAP turėtų standartinį nenutrūkstamą ryšį (angl. *Reliable Messaging*) tarp žinučių bendravimo, tai nereiškia, kad to negalima padaryti su REST interneto paslauga. Panašiai kaip SOAP interneto paslaugų projektuotojai pasirenka, kokią SOAP WS-* specifikaciją naudoti interneto paslaugai įgyvendinti, taip ir REST interneto paslaugos projektuotojai gali suprojektuoti reikiamą funkcionalumą patikimam ryšiui užtikrinti, bet jau ne industrijos standartų grįstu būdu (Hinchcliffe 2005). Šis specifinis įgyvendinimas REST interneto paslaugai nėra idealus, bet svarbiausia, kad REST interneto paslaugų įgyvendinimas gali taip pat būti išplėstas, tik ne pagal reglamentuojamus standartus. Be to, praktikoje ir SOAP žinutės dažnai dėl neteisingo įgyvendinimo, projektavimo ar kitų priežasčių neišnaudoja sukurto saugaus mechanizmo galimybių (Snell 2004).

Microsoft.NET priemonės taip pat naudoja HTTP GET sąsajas interneto paslaugoms realizuoti, o tai labiau panašu į REST metodą. Netgi naujausia SOAP specifikacija (1.2) jau palaiko HTTP GET stiliaus paslaugos alternatyvą, o tai nelabai dera su SOA (O'Neill 2006).

Išvados

1. Išnagrinėjus šiuos du interneto paslaugų įgyvendinimo būdus galima teigti, kad, norint įgyvendinti interneto paslaugas, geriau naudoti REST architektūrinį stilių negu SOAP. Įgyvendinti interneto paslaugas su SOAP yra sudėtingiau nei su REST. Norint supaprastinti interneto paslaugos kūrimą ir projektavimą, reikia naudoti papildomas priemones. SOAP turi paskelbtas specifikacijas, kurias reikia naudoti, bet net iki šiol ne visos yra ištobulintos, pavyzdžiui, WSDL. Programuotojai, naudodami šias specifikacijas SOAP interneto paslaugoms įgyvendinti, gali tikėtis perdaryti kai kurias sistemos dalis, kai pasikeis specifikacijos reikalavimai. Tinklo saugumas gali būti apeitas naudojant SOAP, nes pati SOAP žinutė laiko visą informaciją operacijai atlikti. Būtent dėl to sistemos administratoriams sunkiau stebėti ir apsaugoti sistemą, nepaisant to, kad žinutės saugios, jų turinys gali būti ir pavojingas.
2. Remiantis atlikta analize, tapo aišku, kad tam tikri veiksmai turi būti atlikti, jei norima, kad REST architektūros stilius, įgyvendinant interneto paslaugas, klestėtų ir įgytų pagreitį. Pirma, daugiau dėmesio turi būti skiriama apibrėžiant nuoseklius ir standartinius metodus tam tikram funkcionalumui atlikti, pavyzdžiui, nenutrūkstamo ryšio palaikymas naudojant REST. Specifikacijose, kurias naudoja SOAP, tai jau aprašyta, todėl REST reikia kažko panašaus, tačiau nesudėtingo (kad nuoseklūs ir standartiniai metodai netaptų tokie komplikuoti kaip SOAP). Antra, projektuotojai ir programuotojai turi išmokti naudoti REST sprendimus, kurie remiasi išteklių manipuliavimu per URI, o ne kreipimusi į API, kaip yra su SOAP interneto paslaugomis.

Literatūra

Comerford, C. 2010. *The Security Nightmare of REST Web Services*, February 25 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://features.techworld.com/security/3213655/the-security-nightmare-of-rest-web-services/

Hinchcliffe, D. 2005. *REST vs. SOAP: The Battle of the Web Service Titans*, SOA Web Services Journal, April 26 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://zgia.net/?p=30

Kay, R. 2007. *Representational State Transfer (REST)*, Computer World, August 6 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www.computerworld.com/s/article/ 297424/ Representational_State_Transfer_REST_

O'Neill, M. 2006. *Security for REST Web Services*, RSA Conference, February 14 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www.vordel.com/downloads/rsa_conf_2006.pdf

Prescod, P. 2002. *REST and the Real World*, O'Reilly xml.com, February 20 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www.xml.com/pub/a/ws/2002/02/20/rest.html

Smith, R. 2008. *A Simpler Approach To SOA*, Information Week, August 9 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www.informationweek.com/news/software/soa_webservices/showArticle.jhtml?articleID=209904293

Snell, J. 2004. *Resource-oriented vs. Activity-oriented Web Services*, IBM developerWorks, October 12 [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www-128.ibm.com/developerworks/webservices/library/ws-restvsoap

Tyagi, S. 2006. *RESTful Web Services*, Sun Developer Network, August [žiūrėta 2012 m. rugsėjo 15 d.]. Prieiga per internetą: http://www.oracle.com/technetwork/articles/javase/index-137171.html

COMPARISON OF THE REST ARCHITECTURAL STYLE WITH SOAP IN IMPLEMENTATION OF MODERN WEB SERVICES

K. Stankevičius

Abstract

One of the most relevant steps in the development of an information system during implementation of web services is moving the existing program functionality onto the web. Although web services have been advancing together with software, implementation requires special attention. Accessible options allow organisations to search for alternatives that would suit their needs, i.e. would be cheaper and more effective as well as possibly easier and faster to implement. The paper aims to compare a standard method for creation of web services with another architectural style for implementation of web services, which is getting more widely used.

Keywords: web services, SOAP, service oriented architecture, REST, architectural style.