Web Services

Martin Kuba, ÚVT MU

Zhruba od roku 2000 sílí povyk okolo Web Services. Firma Microsoft na nich založila svoji novou architekturu .NET, firma IBM je označuje za revoluci v e-business, firma SUN hovoří o nové generaci distribuovaných systémů, vznikají o nich specializované časopisy jako *XML & WebServices* a regály knihkupectví se plní tlustými knihami s názvy typu *Professional XML Web Services* a *XML and Web Services Unleashed*.

V tomto komerčním halasu by mohlo zaniknout, že se skutečně jedná o velmi užitečnou technologii, která by mohla změnit způsob, jakým je Internet používán.

Podívejme se, o co vlastně jde. Anglický pojem *Web Services* znamená česky „webové služby“, a je daný historicky, ale dnes je vlastně nesprávný, protože nejde *jen* o služby poskytované přes web. Navíc hrozí záměna se stejným souslovím, jímž firmy vytvářející WWW stránky nebo poskytující prostor k umístění WWW stránek popisují svoji činnost. Nějaký překlad však potřebujeme, používejme tedy pojem *webové služby* jako terminus technicus.

Webové služby jsou novou reinkarnací technologií pro vzdálené volání funkcí v distribuovaných systémech, jako jsou RPC, CORBA nebo RMI[1]. Na rozdíl od jejích předchůdkyň je technologie webových služeb méně vyzrálá (neřeší zatím např. autentizaci a bezpečnost), ale je jednoduchá, otevřená (čti: založená na standardech W3C[2], nikoliv na soukromém standardu jedné firmy či sdružení firem), zcela nezávislá na platformě (operačním systému, programovacím jazyku, typu procesoru), a prochází bouřlivým rozvojem.

Hlavní nadějí vkládanou do webových služeb je, že se WWW změní ze současného souboru HTML stránek srozumitelných pouze lidem[3] na soubor XML stránek (statických i dynamických) čitelných programy a tedy, že programy na zcela různých platformách (JavaScript, Java, C, MS .NET, mobilní telefony) budou moci spolu snadno komunikovat.

Technologii webových služeb tvoří tři části:

- *protokol pro vzdálené volání procedur*, zvaný SOAP, přenášející data zapsaná jako XML
- *jazyk pro popis poskytovaných služeb*, zvaný WSDL
- *mechanismus pro nalezení služeb*, kde spolu soupeří standardy zvané UDDI a WSIL

Podívejme se na ně pěkně popořádku.

1 SOAP

SOAP (Simple Object Access Protocol – česky *jednoduchý protokol pro přístup k objektům*) lze asi nejlépe vysvětlit na postupu, jak historicky vznikl. Už od počátku WWW (okolo roku 1993) bylo možné zavolat program na webserveru a předat mu textové parametry jednoduše tak, že se na konec URL označujícího program přidal znak ? a za něj se uvedly názvy parametrů a jejich hodnoty oddělené znaky &, například takto:

```
http://nekde.cz/cgi/p.exe?param1=hodnota1&p2=h2
```

Tento způsob má jedno omezení, a tím je maximální délka URL, činící v praxi 4kB. Proto byla vymyšlena tzv. metoda POST[4] protokolu HTTP[5], která parametry předává v těle HTTP požadavku; uveďme si, jak takové volání vypadá:

```
POST /cgi/p.exe HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 21

param1=hodnota1&p2=h2
```

Metodou POST je možné předávat jakákoliv data jakékoliv délky, standardizován byl ale jen typ zvaný `application/x-www-form-urlencoded`, jehož tvar je shodný s tvarem parametrů předávaných přímo v URL.

Postupem času (okolu roku 1997) začaly prohlížeče podporovat i typ `multipart/form-data`,

[1] RPC – Remote Procedure Call, CORBA – Common Object Request Broker Architecture, RMI – Remote Method Invocation

[2] W3C – World Wide Web Consortium – organizace definující standardy Internetu

[3] Obsahu HTML stránky rozumí pouze živý člověk. Program vidí jen soubor typograficky upravených textů, najde např. co je napsáno kurzívou, ale nerozumí *smyslu*, nemůže např. najít ve stránce cenu popisovaného výrobku.

[4] od anglického *post* – „poslat“, předchozí metoda volání URL se jmenuje GET – anglicky „získat“.

[5] Hyper Text Transfer Protocol – protokol pro přenos souborů, název je zavádějící, lze přenášet cokoliv, ne jen hypertext

který umožňuje k textovým parametrům přidat obsah souborů.

Vždy bylo možné si metodou POST poslat jakákoliv binární data (například firma SUN doporučovala posílat si tak instance Java objektů mezi Java appletem a Java servletem), ale to předpokládalo mít kontrolu nad *oběma* konci HTTP spojení, a tento požadavek není v obecných aplikacích splněn.

S příchodem jazyka XML, který umožňuje zapsat libovolně složitě strukturovaná data do textového souboru platformově nezávislým způsobem, bylo jen otázkou času, než někoho napadlo posílat si metodou POST data v XML. XML má tu výhodu, že se předávaná data nemusí omezovat na text, je možné předávat si složité objekty a kolekce objektů.

Při použití XML Schema[6] lze navíc jednotným, rozšiřitelným a srozumitelným způsobem popsat strukturu a typy předávaných dat. Použitím XML Namespaces[7] lze snadno zamezit kolizím stejných jmen pro různé věci. Pak již stačilo přidat k předávaným parametrům i informaci jakou funkci je třeba zavolat, a protokol pro vzdálené volání funkcí byl na světě.

1.1 Vývoj standardu

Nejdříve na takovém protokolu začala pracovat firma Microsoft v roce 1998, která ho nazvala SOAP, k ní se poté připojilo mnoho dalších firem včetně IBM. Jiné firmy vyvinuly podobné protokoly s názvy jako XML-RPC, WDDX a XMI. W3C konzorcium vzalo SOAP 1.1 na vědomí[8] v květnu 2000 a vzápětí ustavilo pracovní skupinu zvanou *XML Protocol Working Group* pro vývoj sjednocujícího protokolu pracovně nazvaného XMLP. Tato skupina vyprodukovala protokol nakonec nazvaný SOAP 1.2, který má od 19. 12. 2002 status kandidáta na doporučení.[9]

V současné době existuje několik desítek různých implementací SOAP na nejrůznějších platformách, od C/C++, přes Javu, .NET, Perl až po JavaScript v prohlížeči Mozilla. Jejich vzájemná kompatibilita je pravidelně testovaná sadou testů *SOAPBuilders Interoperability Lab*.

Původní nápad přenášet SOAP zprávy protokolem HTTP byl později rozšířen na možnost přenosu jinými přenosovými protokoly, například SMTP (Simple Mail Transport Protocol), tedy elektronickou poštou, čímž slovo Web v názvu Web Services pozbylo trochu smysl, nicméně zůstalo zachováno.

1.2 Příklad volání

Uveďme si příklad, jak takové SOAP volání vlastně vypadá. Mějme například funkci `boolean jePrvocislo(long cislo)`, která má číselný parametr jménem `cislo` a vrací pravdivostní hodnotu podle toho, zda je parametr prvočíslo nebo ne. Vzhledem k tomu, že webová služba může být implementovaná v jakémkoliv programovacím jazyce, typy `long` a `boolean` jsou typy definované v XML Schema a ne nutně přítomné v daném programovacím jazyku.

SOAP vyžaduje, aby jméno každé funkce bylo v určitém jmenném prostoru. U objektově orientovaných jazyků tento jmenný prostor odpovídá objektu a funkce jeho metodě, u implementace třeba v jazyku C tento jmenný prostor přímý smysl nemá, ale je nutné nějaký prostor definovat. Jmenný prostor je určen nějakým URI[10], zvolme třeba `urn:mojeURI`. Příklad SOAP zprávy volající takovou funkci je na obrázku č.1.

Tato zpráva je přenesena na server, který funkci implementuje, obvykle metodou POST protokolu HTTP, nebo emailem či jiným způsobem. Postup, jímž je na serveru nalezena funkce, která má být zavolána, záleží na implementaci serveru, obvykle je funkce určena svým jménem a jmenným prostorem. Další možnosti jsou: URL na které přišel HTTP požadavek, obsah speciální HTTP hlavičky `SOAPAction`, uživatelem definovaný obsah tagu `env:Header`, nebo něco jiného. SOAP je

[6] XML Schema – jazyk pro definice struktury a typovosti dat, viz článek *XML snadno a rychle*

[7] XML Namespaces – mechanismus pro rozlišení různých množin tagů v jednom dokumentu

[8] SOAP 1.1 má status *W3C Note* (poznámka), což znamená pouhé zveřejnění a žádnou podporu ze strany W3C

[9] Vývoj standardu W3C probíhá ve čtyřech fázích: první je *W3C Working Draft* (pracovní návrh), následuje *W3C Candidate Recommendation* (kandidát na doporučení) – návrh určený k veřejné diskuzi, třetí fáze je *W3C Proposed Recommendation* (navržené doporučení), a hotový standard se nazývá *W3C Recommendation* (doporučení).

[10] Uniform Resource Identifier, URI jsou sjednocením URL a URN – Uniform Resource Names

```
<env:Envelope
  xmlns:env=„http://schemas.xmlsoap.org/soap/envelope/“
  env:encodingStyle=„http://schemas.xmlsoap.org/soap/encoding/“
  xmlns:xs=„http://www.w3.org/1999/XMLSchema“
  xmlns:xsi=„http://www.w3.org/1999/XMLSchema-instance“>
 <env:Header/>
 <env:Body>
  <m:jePrvocislo xmlns:m=„urn:mojeURI“>
    <cislo xsi:type=„xs:long“>1987</cislo>
  </m:jePrvocislo>
 </env:Body>
</env:Envelope>
```

Obrázek 1: Příklad SOAP zprávy pro volání funkce.

```
<env:Envelope
  xmlns:env=„http://schemas.xmlsoap.org/soap/envelope/“
  xmlns:xsi=„http://www.w3.org/1999/XMLSchema-instance“
  xmlns:xsd=„http://www.w3.org/1999/XMLSchema“>
 <env:Body>
  <ns1:jePrvocisloResponse
    xmlns:ns1=„urn:mojeURI“
    env:encodingStyle=„http://schemas.xmlsoap.org/soap/encoding/“>
   <return xsi:type=„xsd:boolean“>true</return>
  <ns1:jePrvocisloResponse>
 <env:Body>
<env:Envelope>
```

Obrázek 2: Příklad SOAP zprávy vracející výsledek.

v tomto benevolentní a nepředepisuje jeden povinný způsob.

Všimněme si, že v XML představujícím toto SOAP volání jsou zavedeny čtyři prefixy jmenných prostorů: `env` použitý pro tagy samotného SOAP, `xs` pro XML Schema, použitý v označení typu `xs:long`, `xsi` pro instanci XML Schema, použitý pro odlišení atributu `xsi:type` určujícího typ parametru funkce a konečně námi definovaný `m` použitý pro označení naší funkce. Jak vyplývá z definice XML Namespaces, samotné řetězce použité jako prefixy nejsou důležité, a mohly být zvoleny libovolné jiné, důležité jsou jen URI, ke kterým se prefixy vážou. Například místo prefixu `env` mohl být použit prefix `SOAP-ENV`, nebo `krteček`, výsledek by to nezměnilo.

2 WSDL

Možnost vzdáleně volat funkce pomocí SOAP je k ničemu, pokud nevíme, jaké funkce se dají zavolat, jaké mají parametry a jaké vrací hodnoty. Tento problém řeší jazyk WSDL (Web Services Description Language, česky *jazyk popisu webových služeb*), založený na XML a hojně využívající standardy XML Namespaces a XML Schema. Vznikl sloučením tří jazyků firem IBM, Microsoft a Ariba s názvy NASSL, SCL a SDL do jednoho jazyka nazvaného WSDL 1.1. W3C vzalo WSDL 1.1 na vědomí a ustavilo pracovní skupinu s názvem *Web Services Description Working Group*, která nyní pracuje na jeho novější verzi WSDL 1.2 (zatím je ve fázi pracovního návrhu).

2.1 Struktura WSDL dokumentů

WSDL jazyk je velice obecný, aby zachoval platformovou nezávislost. Popišme si, jak takový popis webové služby ve WSDL vypadá. WSDL dokument obsahuje popis jedné *služby* (klíčové slovo `service`, viz obrázek č.3), což je největší jednotka. Jedna služba má jednu nebo více *bran* (`port`). Každá brána má *vazbu* (`binding`), což je způsob jak se daná brána volá (například SOAP-přes-HTTP), a nějakou přístupovou adresu (URL). Lze tedy teoreticky mít pro jednu službu více bran s různými vazbami, tj. volat jednu službu

```
<message name=„jePrvocisloRequest“>
 <part name=„cislo“ type=„xsd:long“/>
</message>
<portType name=„Cisla“>
 <operation name=„jePrvocislo“ parameterOrder=„cislo“>
  <input message=„m:jePrvocisloRequest“ name=„jePrvocisloRequest“/>
  <output message=„m:jePrvocisloResponse“ name=„jePrvocisloResponse“/>
 </operation>
</portType>
<binding name=„cislaSoapBinding“ type=„m:Cisla“> ... </binding>
<service name=„CislaService“>
 <port binding=„m:cislaSoapBinding“ name=„cisla“>
  <wsdlsoap:address location=„http://nekde.cz/cisla“/>
 </port>
</service>
```

Obrázek 3: Část WSDL souboru definující funkci `jePrvocislo`

různými způsoby, např. první SOAP-přes-HTTP, druhou SOAP-přes-HTTP-nad-SSL a třetí SOAP-přes-SMTP. Prakticky budou fungovat jen první dvě, protože WSDL 1.1 má definovánu syntaxi jen pro vazby založené na HTTP.

Vazby odkazují na *rozhraní* (`portType`), které je souhrnem *operací* (`operation`). Rozhraní v objektově-orientovaných jazycích odpovídá objektu, jednotlivé operace odpovídají metodám objektu, nebo v jazyce C funkcím.

Každá operace definuje obvykle dvě *zprávy* (`message`), jednu vstupní a jednu výstupní, ale může i míň. Každá zpráva obsahuje žádnou, jednu nebo více *částí* (`part`), které odpovídají parametrům a návratovým hodnotám. Z toho plyne, že volané funkce mohou mít více návratových hodnot než jen jednu!

Typy parametrů a návratových hodnot jsou definovány pomocí XML Schema. Jako jednoduché typy jsou tedy k dispozici řetězce (`string`), čísla s pohyblivou (`float`) i pevnou (`decimal`) řádovou čárkou, pravdivostní hodnoty (`boolean`), binární data (`base64Binary`), časový okamžik (`dateTime`), časový interval (`duration`), URL (`anyURI`); dále jejich odvozeniny – výčtové typy, číselné intervaly, záporná čísla (`negativeInteger`), celá čísla různého rozsahu (`int, byte, short, long`). Je možné definovat složené typy vzniklé jako souhrny nebo varianty z jiných typů, jednoduchých i složených, dokonce lze definovat jeden typ jako rozšíření druhého, lze tedy vyjádřit dědičnost objektů.

2.2 Nástroje pro práci s WSDL

Dokument v jazyce WSDL příslušný k určité webové službě ji plně popisuje, pokud tedy máme WSDL popis, můžeme webovou službu začít používat. Dokonce existují automatizované nástroje, které z WSDL popisu vygenerují kód pro volání služby v nějakém konkrétním programovacím jazyce, a tento kód pak lze volat stejným způsobem jako by se jednalo o lokálně implementované funkce (takovému zástupnému kódu se říká *proxy* nebo *stub*).

Dále ani není nutné WSDL dokumenty psát ručně, existují nástroje pro jejich generování přímo z kódu programovacího jazyka. Lze tedy s výhodou použít postup, kdy se nejdřív popíše jen rozhraní volatelných funkcí (vytvoří se jen `interface` v jazyce Java nebo deklarace funkcí v `.h` souboru v jazyce C), automatizovaným nástrojem se vygeneruje WSDL popis zatím neexistující webové služby, dalším automatizovaným nástrojem se z WSDL dokumentu vygeneruje kód pro volání i implementační serverovou část, a stačí pak dopsat vlastní funkcionalitu do vygenerovaného kódu.

3 UDDI a WSIL

Zatímco SOAP a WSDL jsou zavedené a v praxi používané standardy, v oblasti nalézání webových služeb zatím není rozhodnut boj o to, jaký mechanismus se bude používat.

Historicky starší UDDI (Universal Description, Discovery and Integration, česky *univerzální popis, objevování a spojování*) nabízí veřejnou databázi (anglicky *registry*), do které poskytovatelé webových služeb mohou ukládat popisy služeb a uživatelé ji prohledávat. Dvě takové centrální databáze spravují firmy IBM a Microsoft. Praxe ale ukázala, že plné dvě třetiny záznamů v těchto databázích jsou neplatné. Druhý a hlavní problém je ten, že databáze nijak nezaručuje důvěryhodnost poskytovatelů služeb. Představte si například, že si chcete založit bankovní účet a ovládat ho jako webovou službu. Vyberete si poskytovatele ve veřejné databázi, do které může kdokoliv zapisovat a svěříte mu své peníze ? Rozhodně ne.

Uděláte to opačně – nejdřív si najdete poskytovatele služby, kterému věříte, a požádáte ho o popis rozhraní. Tímto mechanismem řeší nalézání WSIL (Web Services Inspection Language, česky *jazyk přehlídky webových služeb*), mimochodem také vyvinutý firmami IBM a Microsoft. WSIL popisuje služby poskytované nějakým poskytovatelem pomocí souboru jménem `inspection.wsil` umístěného vždy v hlavním adresáři webserveru poskytovatele. Tento popis je tedy na všeobecně známém místě, a prohledávací služby jako Google a Yahoo mohou nabízet vyhledávání v těchto souborech.

Ani UDDI, ani WSIL nejsou standardy W3C konzorcia, ani neexistuje v rámci W3C pracovní skupina, která by se jim věnovala, lze tedy říci, že tyto technologie ještě dostatečně nedozrály.

4 Nástroje pro implementaci

Nástrojů pro implementaci webových služeb je k dispozici hodně, komerčních i freewareových. Obvykle zahrnují generátor WSDL, generátor kódu pro klienta služby, generátor kódu pro serverovou implementaci služby, nějaký aplikační server ve kterém služba běží a komerční implementace i vývojové prostředí s GUI, kde si lze vše naklikat myší.

Z free nástrojů je asi nejpoužívanější *Apache Axis*, implementace v jazyku Java, používající jako aplikační server libovolný Java servlet container[11]. Zahrnuje nástroj `Java2WSDL` pro generování WSDL ze zdrojových kódů v Javě, i nástroj `WSDL2Java` pro generování klientských i serverových zdrojových kódů z WSDL.

Asi nejrychlejší webové služby generuje freewareový gSOAP, generátor implementací pro C/C++. Zatímco Axis dokáže zavolat libovolnou webovou službu pomocí dynamicky zkonstruovaného SOAP volání, gSOAP vygeneruje „natvrdo“ jednoúčelový kód pro volání konkrétní webové služby a obecnými věcmi se nezdržuje. Výměnou za menší flexibilitu je jeho extrémní rychlost. Obsahuje generátor WSDL z C/C++ kódu i generátor C/C++ kódu z WSDL.

Mezi producenty komerčních nástrojů září pražská firma Systinet s implementacemi pro Javu i C++, což dokazuje, že se česká firma dokáže udržet na špici světového vývoje. Dalšími velkými hráči jsou IBM (WebSphere), Microsoft (.NET) a Oracle (Oracle9iAS).

5 Budoucnost a perspektivy

Webové služby mají velký potenciál. Už dnes umožňují, aby se informační systémy různých institucí snadno volaly navzájem, přestože jsou každý založen na úplně jiné platformě. Představte si na chvíli, že by státní úřady začaly poskytovat své služby jako webové služby. Když by někdo chtěl stavební povolení, zavolal by funkci `vydejPovoleni()` systému na stavebním úřadě, ten by automaticky zavolal informační systém katastrálního úřadu a zjistil kdo je vlastník, poté by se zeptal systému památkového úřadu, zda je to chráněná památka, systému hasičů by se zeptal jestli nemá námitky, obratem by vydal povolení a zavolal službu České Pošty, aby povolení vytiskla na papír a doručila (ten papír je nutný, povolení se musí vystavit za okno :-). Krásná představa ...

Odvážil bych se říci, že webové služby jsou dnes v podobném stavu, jako byl World Wide Web dejme tomu v roce 1994 – většina technologie je

[11] Java servlet container – v prostředí Javy obdoba webserveru spouštějícího programy, místo spouštění procesů ale volá metody Java objektů. Příkladem jsou Apache Tomcat nebo Jetty.

hotova, dá se to používat, ale ještě zbývá dotáhnout do konce některé věci a hlavně jejich existence musí vejít do obecného povědomí.

6 Odkazy

- http://www.w3.org/2002/ws/ – W3C Web Services Activity
- http://www.w3.org/TR/SOAP/ – definice SOAP
- http://www.w3.org/TR/wsdl – definice WSDL
- http://www.w3.org/TR/xmlschema-2/#built-in-datatypes – typy definované XML Schema
- http://www.uddi.org/ – definice UDDI
- http://www.ibm.com/developerworks/library/ws-wsilspec.html – definice WSIL
- http://xml.apache.org/axis/ – Apache Axis
- http://www.xmethods.com/ – seznam veřejných služeb pro pokusy □