UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Vytěžování informací o počítačích pomocí Windows Management Instrumentation

Ondřej Barák

Bakalářská práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Ondřej Barák**
Osobní číslo:	**I10004**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Vytěžování informací o počítačích pomocí WMI**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Autor bakalářské práce popíše Windows Management Instrumentation (WMI) a jeho použití v podnikových sítích z hlediska správy a monitoringu dané sítě. V praktické části autor nainstaluje Windows Server 2008 a vytvoří modelovou síť a naprogramuje sledovací systém pro vytěžování informací o počítačích pomocí WMI.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

***JONES, Don. Managing Windows with VBScript and WMI. Boston: Addison-Wesley, c2004, xxxvii, 601 p. Microsoft Windows server system series. ISBN 03-212-1334-3.**
***WILSON, Ed. Microsoft Windows scripting with WMI: self-paced learning guide. Redmond, Wash.: Microsoft Press, c2006, xxvii, 370 p. ISBN 978-073-5622-319.**
***http://msdn.microsoft.com/en-us/library/windows/desktop/aa384642%28v=vs.85%29.aspx**

Vedoucí bakalářské práce: **Ing. Soňa Neradová**
Katedra softwarových technologií

Datum zadání bakalářské práce: **21. prosince 2012**
Termín odevzdání bakalářské práce: **10. května 2013**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 29. března 2013

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 10. 5. 2013 Ondřej Barák

Poděkování

Na tomto místě bych rád poděkoval Ing. Soně Neradové za zajímavé téma, umožnění přístupu do síťové laboratoře a cenné rady při vypracovávání této práce.

Anotace

Cílem této práce je popsat Windows Management Instrumentation a jeho využití ve správě počítačových sítí. Teoretická část se zabývá popisem technologie WMI a principem jeho funkčnosti. Dále bude představen nástroj PowerShell, který v kombinaci s WMI představuje mocný prostředek pro správu a sledování sítě. Praktická část je zaměřena na vytvoření programu, který bude s pomocí WMI získávat informace o počítačích.

Klíčová slova

WMI, WQL, CIM, PowerShell, skript, počítačová síť

Title

Extraction of information about computers with the use of Windows Management Instrumentation

Annotation

The aim of this paper is to describe the Windows Management Instrumentation and its use in computer network administration. The theoretical part deals with the description of the WMI technology and the principles of its functionality. Furthermore, the PowerShell tool is introduced, which in combination with WMI presents a powerful means to network administration and monitoring. The practical part focuses on creating a programme which will be able to get information about computers with the help of WMI.

Keywords

WMI, WQL, CIM, PowerShell, script, computer network

Obsah

Seznam zkratek

API	Application Programming Interface
CIM	Common Information Model
CIMOM	Common Information Model Object Manager
CMIP	Common Management Information Protocol
COM	Component Object Model
DCOM	Distributed Component Object Model
DHCP	Dynamic Host Configuration Protocol
DLL	Dynamic Link Library
DMTF	Distributed Management Task Force
DNS	Domain Name Systém
IPX	Internetwork Packet Exchange
PHP	Hypertext Preprocessor
PERL	Practical Extraction and Reporting Language
SDK	Software Development Kit
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol
VBScript	Visual Basic Script Edition
WAN	Wide Area Network
WBEM	Web-Based Enterprise Management
WMI	Windows Management Instrumentation
WQL	Windows Management Instrumentation Query Language

Seznam obrázků

Seznam tabulek

Úvod

V dnešní době jsou počítačové sítě nezbytnou součástí mnoha organizací. Zajištění jejich bezproblémového chodu je tedy pro ně velmi důležité. S rostoucí velikostí a složitostí sítě se samozřejmě zvyšují nároky na její správu. V malých podnicích s několika počítači může správce řešit administrační záležitosti a vzniklé problémy na každé stanici individuálně. Jestliže ale daná organizace vlastní řádově desítky nebo stovky počítačů umístěných v různých budovách, stane se řešení v podobě kontroly každé stanice zvlášť časově neefektivní. Tento problém se sice dá vyřešit zaměstnáním více lidí, starajících se o síť, avšak u tohoto řešení převahují zápory nad klady. Další možností je využít nástroj, určený k řešení tohoto typu problému a potřebné činnosti automatizovat. Toto řešení je mnohem efektivnější, protože lze úkon provést v neporovnatelně kratším čase a bez potřeby najímat více lidí. Tato práce se zabývá právě takovým nástrojem, který umožňuje správu a sledování sítě postavené na platformě Microsoft Windows. Windows Management Instrumentation byla po svém vydání dlouho a neprávem opomíjená technologie. V současnosti se mu však dostává stále více pozornosti.

V teoretické části bude popsán princip funkčnosti a základní součásti WMI. Budou zmíněny standardy s touto technologií související. Dále bude ve stručnosti popsán nástroj PowerShell, který bude v poslední kapitole použit jako prostředek pro práci s WMI.

Cílem praktické části je vytvořit nástroj na sledování počítačů v síti. Aplikace bude vytvořena pomocí PowerShellu. Tento demonstrační program se bude moci připojit ke vzdálenému počítači a zobrazit o něm informace.

1 Windows Management Instrumentation

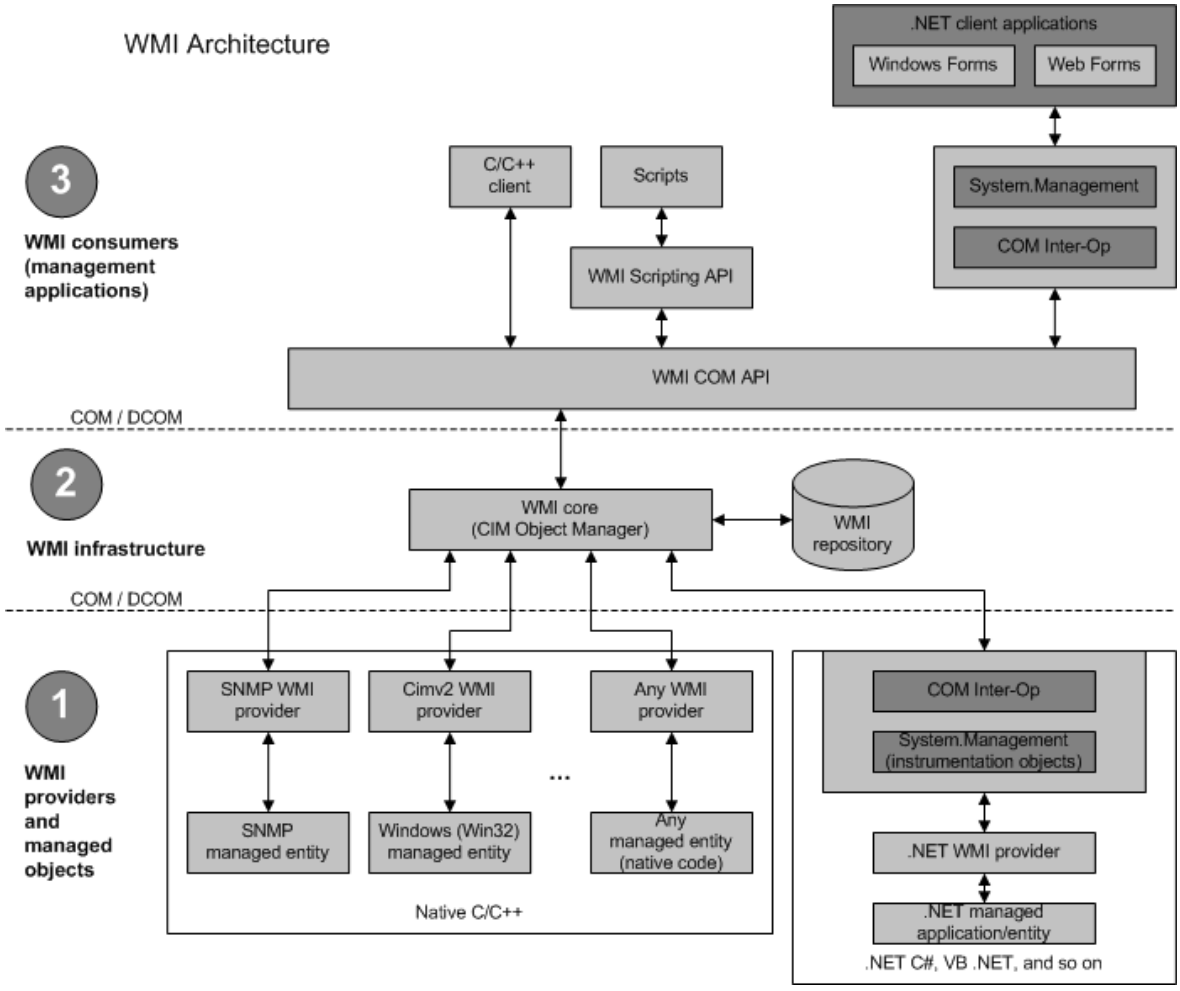
Společnost Microsoft vyvinula WMI jako svoji vlastní implementaci iniciativy WBEM, což je množina standardů, které byly vytvořeny za účelem sjednotit postupy při správě rozlehlých počítačových sítí. Za touto iniciativou stojí sdružení DMTF, jehož je Microsoft členem. WMI je tedy s WBEM kompatibilní a poskytuje integrovanou podporu pro datový model CIM.

WMI lze využít nejen k získávání velmi detailních informací o počítačích ale i k jejich konfiguraci. Dovede také spolupracovat s dalšími službami pro správu v systému Windows. Prostřednictvím rozhraní API lze vytvářet aplikace pro správu. K WMI lze přistupovat z různých programovacích jazyků – C/C++, C#, Microsoft Visual Basic, VBScript, PowerShell a další. WMI obsahuje také vlastní dotazovací jazyk WMI Query Language, se kterým lze přistupovat ke spravovaným objektům.

WMI se poprvé objevilo v roce 1998 jako přídavek pro Windows NT 4, avšak mnohem větší podpory se mu dostalo až od Windows 2000, ve kterém je už od základu nainstalováno. Vývoj WMI samozřejmě neustává a každá další verze Windows rozšiřuje jeho možnosti nebo mění funkčnost stávajících.

1.1 Architektura WMI

WMI poskytuje jednotné rozhraní pro všechny lokální nebo vzdálené aplikace a skripty, které získávají data z počítačových systémů a sítí. Slouží tedy jako abstraktní vrstva mezi aplikacemi pro správu a fyzickými a logickými prostředky, které spravují. Toto rozhraní je vytvořeno tak, aby klientské WMI aplikace a skripty nemusely volat širokou škálu aplikací operačního systému a programovacích rozhraní aplikací. K získání dat z WMI stačí vytvořit aplikaci nebo napsat skript, který z WMI tříd získá potřebné informace. Skládá se z 3 hlavních částí – WMI spotřebitelé, WMI infrastruktura a spravovaných objektů. Následující obrázek (Obrázek 1) demonstruje rozložení a vztahy mezi zmiňovanými komponentami.



Obrázek 1 - Architektura WMI [1]

1.1.1 WMI poskytovatelé a spravované objekty

WMI poskytovatel je COM/DCOM objekt, který sleduje jeden nebo víc spravovaných objektů. Poskytovatelé dále zásobují WMI daty ze spravovaných objektů a pracují se zprávami zaslanými WMI těmto objektům. Dále komunikují s CIMOM pomocí programovacích rozhraní WMI. Spravované objekty jsou logické nebo fyzické části systému (například procesor, souborový systém, procesy a mnoho dalších), které jsou modelovány pomocí modelu CIM. Tabulka 1 uvádí rozdělení a popis jednotlivých druhů poskytovatelů.

Tabulka 1 - Rozdělení a popis poskytovatelů [1]

Klasifikace	**Popis**
Třída	Poskytuje aplikacím definice tříd.
Instance	Poskytuje WMI třídy.
Vlastnost	Poskytuje různé způsoby práce s hodnotami vlastností objektu.
Metoda	Poskytuje metody pro třídy specifických poskytovatelů.
Událost	Generuje oznámení o události.
Událost spotřebitele	Poskytovatel určující, který spotřebitel zpracuje danou událost.

1.1.2 Infrastruktura WMI

Základním kamenem infrastruktury WMI je CIMOM, který zprostředkovává komunikaci mezi poskytovateli a spotřebiteli. Všechny požadavky od WMI jdou skrze CIMOM. CIMOM dále poskytuje některé služby jako například vzdálený přístup, zpracování událostí a podobně. Druhou částí WMI infrastruktury je WMI repositář, který slouží jako databáze ve které jsou uloženy objekty tříd a statická data o systému.

1.1.3 WMI spotřebitelé

Spotřebitel může být aplikace pro správu, skript nebo třeba webová aplikace, která zpřístupní a zobrazí nebo zpracuje data o spravovaných objektech. Příkladem spotřebitele je nástroj vytvořený jako praktická část této práce.

V Oddílu 1.1 bylo čerpáno z [1], [2], [3].

1.2 Jmenné prostory

Tyto logické celky obsahují dostupné WMI třídy a jejich instance. Jsou uspořádány ve stylu stromové struktury. Kořenový jmenný prostor se nazývá root a umístění každého jmenného prostoru je popsáno cestou stejně jako v adresářové struktuře (např. root\cimv2). Všechny třídy v jednom daném jmenném prostoru musí mít jedinečný název, ale je možné mít dvě odlišné třídy se stejným názvem, avšak každá musí být umístěna v jiném jmenném prostoru. Dále musí platit, že odvozená a rodičovská třída musí sdílet stejný jmenný prostor. WMI obsahuje všechny třídy dostupné v modelu CIM, které jsou dostupné ve jmenném prostoru root/cimv2.

V oddílu 1.2 bylo čerpáno z [4].

1.3 Třídy

Třídy reprezentují určitou část systému. Může se jednat o hardware (např. síťová karta), software (např. operační systém), služba jako třeba DNS nebo dokonce úložiště dat (např. registr). Základem pro objektový model WMI je již zmiňovaný CIM, který definuje tři kategorie dostupných tříd:

- **Základní třídy** představují spravované objekty, které se vztahují ke všem oblastem správy.
- **Společné třídy** představují spravované objekty, které se vztahují pouze k určitým oblastem správy.
- **Rozšířené třídy** jsou specifické z pohledu použité technologie. Jedná se například o třídy vytvořené za účelem správy platformy Windows a aplikací společnosti Microsoft.

WMI třídy lze rozdělit do dvou skupin. První skupina je složena ze systémových tříd, které jsou přítomny v každém jmenném prostoru. Poskytují informace o tomto jmenném prostoru, registracích a zabezpečení. Dále jsou zodpovědné za zpracování událostí vytvořených jinými WMI třídami.

Druhá skupina zahrnuje společně základní CIM třídy a rozšiřující třídy vytvořené společností Microsoft. Tyto třídy jsou používané ke správě prostředí Windows.

WMI třídy podporují tři typy atributů:

- **Vlastnosti** jsou charakteristické pro spravované objekty (například jméno počítačového systému)
- **Metody** představují akce, které spravované objekty mohou vykonat (například metoda poskytující možnost zapnutí nějaké služby)
- **Asociace** jsou odkazy na speciální typ WMI tříd, které představují vztah mezi jinými objekty.

V oddílu 1.3 bylo čerpáno z [5], [6].

1.4 Zabezpečení WMI

Vzdálená kontrola počítače a snadné získání informací o něm pomocí WMI ušetří hodně práce, ale bez patřičného zabezpečení by tato technologie byla téměř nevyužitelná. WMI podporuje bezpečnostní mechanismy, které nepřepisují ani neobcházejí zabezpečení operačního systému, ale vytvářejí nad ním další bezpečnostní vrstvu.

Prvním způsobem je možnost omezit přístup k daným jmenným prostorům určitým uživatelům. Tato omezení lze snadno nastavit v nástroji „wmimgmt.msc“. Zabezpečení jmenných prostorů WMI je zajištěno porovnáním přístupového tokenu uživatele, který se k němu chce připojit a popisovačem zabezpečení daného jmenného prostoru. Toto ověření se provádí pouze při připojování uživatele. Uživatelé, kteří se nacházejí ve skupině Administrators mají ve výchozím nastavení ve spravovaném počítači neomezený přístup ke správě služby WMI. Ostatní uživatelé mimo skupinu Administrators mají dovoleno pouze číst, zapisovat a spouštět například služby pouze ve svých místních počítačích. Tabulka 2 uvádí popis a typy oprávnění k přístupu ke jmenným prostorům, které lze uživatelům nastavit.

Tabulka 2 - Druhy a popis oprávnění omezující přístup ke jmenným prostorům [7]

Oprávnění	**Popis**
Plně zapisovat	Povolí úplný přístup pro čtení, zápis i odstraňování ke všem objektům, třídám a instancím služby WMI.
Částečně zapisovat	Povolí přístup pro zápis do statických objektů služby WMI.
Zapisovat poskytovatele	Povolí přístup pro zápis k objektům poskytnutým poskytovatelem.
Spouštět metody	Povolí spouštění metod exportovaných ze tříd nebo instancí služby WMI, které mají být spuštěny.
Povolit účet	Povolí přístup ke čtení objektů služby WMI.
Povolovat vzdálený přístup	Povolí vzdálený přístup ke jmennému prostoru.
Číst zabezpečení	Povolí přístup jen pro čtení k informacím o zabezpečení služby WMI.
Upravovat zabezpečení	Povolí přístup pro čtení a zápis k informacím o zabezpečení služby WMI.

Další možností jsou bezpečnostní nastavení DCOM, který WMI používá ke zpracování volání ze vzdálených počítačů. Tato nastavení lze zobrazit nebo změnit v nástroji „dcomcnfg.exe“ a lze je rozdělit na autentizaci a zosobnění. Autentizace je způsob, kterým se jeden proces identifikuje druhému, zatímco zosobnění indikuje, jak moc velká oprávnění udělí klient serveru k zavolání dalších procesů.

V oddílu 1.4 bylo čerpáno z [8], [9].

1.5 Alternativní technologie pro správu sítě

1.5.1 Simple Network Management Protocol

Tento široce rozšířený protokol je určen zejména ke správě rozsáhlých sítí. Je součástí sady internetových protokolů TCP/IP a jeho první verze se objevila již v osmdesátých letech. Lze ho využít mimo jiné například ke sběru statistických dat o provozu v síti nebo k nastavování různých vzdálených zařízení. Komunikace v SNMP vyžaduje dvě strany:

- **Agent** je umístěn na zařízení (například směrovač, přepínač, VoIP telefon atd.), které je připojené do sítě a sbírá o něm informace, jež dále posílá správci.
- **Správce** má na starosti správu všech příchozích informací, které dostal od agentů na dalších zařízeních v síti.

V současné době je SNMP dostupný ve třech verzích. SNMPv1 a SNMPv2 jsou si velmi podobné, protože obě verze používají stejnou metodu detekce paketů v síti. SNMPv2 má oproti své starší verzi pouze několik novinek, mezi nimiž je například možnost vložit více SNMP požadavků do jednoho paketu. SNMPv3 už přinesla poněkud výraznější změny, zejména co se bezpečnosti týče. V SNMPv3 lze využít autentizaci pomocí uživatelského jména a hesla a šifrování paketů.

1.5.2 Common Management Information Protocol

Protokol CMIP vznikl ke konci osmdesátých let a poskytuje implementaci služeb definovaných CMIS. Jedná se stejně jako SNMP o protokol aplikační vrstvy. Je založen na modelu ISO/OSI a funguje také na podobném principu (model klient/server) jako SNMP. Avšak na rozdíl od SNMP je mnohem komplexnější a poskytuje více možností, zároveň je ale náročnější co se systémových prostředků týče. V dnešní době není tolik rozšířen a je používán zejména v sítích WAN a v telekomunikačních službách.

1.5.3 NetFlow

Za tímto otevřeným protokolem stojí společnost Cisco Systems. Oproti výše zmíněným je zaměřen výhradně na monitorování datového toku procházejícího skrze síťová zařízení. Aby však mohl NetFlow na daném zařízení pracovat, musí na něm být podporován. Získaná data lze exportovat a dále zpracovávat. Z těchto statistik lze detekovat mnoho problémů jako například odhalování úzkých míst v síti, spamu, bezpečnostních incidentů nebo špatně nastavených zařízení. NetFlow je dnes dostupný již ve své deváté verzi a je

považován za standard v oboru sledování provozu v síti. Aby mohla být data zachycena, odeslána a analyzována, je třeba tří základních částí:

- **Exportér** sleduje pakety a vytváří záznamy ze sledovaného provozu. Tato získaná data dále odesílá do sběrače.
- **Sběrač** přijímá data vyslaná exportérem, ukládá je v lokální databázi a posílá záznamy o nich vyhodnocovači.
- **Vyhodnocovač** analyzuje získané záznamy podle aktuální potřeby.

V pododdílu 1.5.3 bylo čerpáno z [10].

1.6 Common Information Model

Jedná se o další standard sdružení DMTF, který vznikl za účelem poskytnutí ucelené definice správy informací pro systémy, sítě, aplikace a služby. To má umožnit jednotnou správu zmíněných prvků bez ohledu na jejich výrobce nebo prodejce. Jednotlivým elementům také dovoluje sdílet mezi sebou informace o správě těchto prvků. CIM nejenže reprezentuje tyto prvky a informace, ale poskytuje i prostředky k jejich aktivní kontrole a správě. CIM je tedy rozšiřitelný, objektově orientovaný a na jazyku nezávislý datový model. WMI obsahuje rozšíření tohoto modelu, které popisuje platformu operačních systému Microsoft Windows.

V oddílu 1.6 bylo čerpáno z [11].

1.7 WMI Query Language

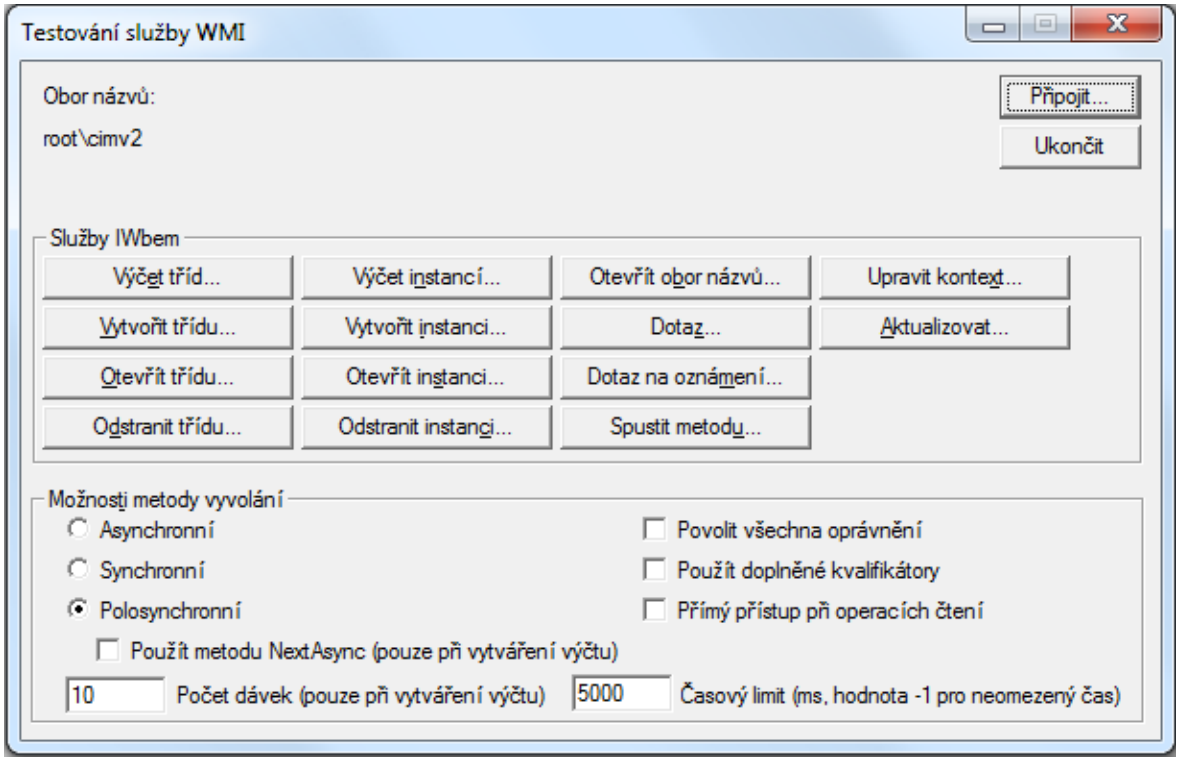
WQL je podmnožina dotazovacího jazyka SQL s drobnými sémantickými změnami. Na rozdíl od SQL však neposkytuje příkazy pro vkládání, mazání nebo změnu dat. WQL dále nepodporuje dotazy, které by zahrnovaly výběr dat z několika jmenných prostorů najednou. Obsahuje ale rozšíření na podporu funkcí specifických pro WMI. Podobně jako SQL obsahuje WQL sadu klíčových slov a operátorů a podporuje tyto tři typy dotazů:

- **Datové dotazy**, které jsou používány nejčastěji a představují jednoduchou formu získávání dat z WMI. Jsou využívány k získání instancí tříd a datových asociací.
- **Schématické dotazy**, jež nacházejí uplatnění v případě, kdy je třeba získat definici třídy a schématické asociace. Používají se hlavně při získávání informací o samotném WMI a jeho struktuře.
- **Událostní dotazy** jsou spotřebiteli využívány k registraci oznámení o událostech. Poskytovatelé událostí pak pomocí těchto dotazů registrují podporu jedné nebo více událostí.

Ukázka jednoduchého WQL dotazu, který vrátí názvy všech běžících procesů začínajících na písmeno „c“:

```
SELECT Name FROM Win32_process WHERE Name LIKE 'c%'
```

Pro testování WQL dotazů lze využít nástroj WBEMTest (Obrázek 2), který je dostupný na počítačích s nainstalovaným WMI.



Obrázek 2 - Nástroj WBEMTest

V oddílu 1.7 bylo čerpáno z [6], [12].

2 PowerShell

PowerShell je objektově založený shell a skriptovací jazyk vytvořený společností Microsoft a je určený zejména ke správě systému. PowerShell je postavený na techologii .NET Framework a dovede proto využívat všechny jeho možnosti. Dále má přístup k objektům poskytovaným COM a WMI, což umožňuje správu jak lokálních, tak vzdálených počítačů. Od ostatních shellů se liší především v objektovém přístupu k datům, místo jejich zpracovávání jako text. PowerShell byl od základu vytvořen jako nový skriptovací jazyk avšak jeho tvůrci se inspirovali dalšími jazyky jako třeba PHP, PERL, Python nebo C#.

První verze PowerShellu byla vydána v roce 2006 pro operační systémy Windows XP SP2/SP3, Windows Server 2003 a Windows Vista. Verze 2.0 vyšla spolu s Windows 7 a Windows 2008 R2 ve kterých je od základu nainstalovaná. Tato verze přinesla výrazné změny a mnoho novinek. Vývoj však pokračuje a s příchodem Windows 8 a Windows Server 2012 se objevila verze 3.0, která přinesla další rozšíření funkčnosti PowerShellu.

2.1 Cmdlety

Cmdlet je v podstatě obdoba příkazů používaných v jiných shellech. Rozdíl je v tom, že cmdlety jsou implementovány za použití .NET tříd, zkompilovány v DLL a nahrány do procesu PowerShell během jeho spouštění. Díky této vlastnosti může kdokoliv využít PowerShell SDK a vytvořit si vlastní cmdlet, a to ve kterémkoliv jazyce, který podporuje .NET.

Funkce, kterou v prostředí PowerShell implementuje cmdlet je velice specifická a většinou zaměřená na určitý druh objektu. Názvy cmdletů jsou proto vytvořeny tak, aby co nejlépe vystihovaly akci, kterou má daný příkaz vykonat. Název je složen ze slovesa a podstatného jména odděleného pomlčkou. Sloveso představuje akci, kterou cmdlet provádí a podstatné jméno zastupuje objekt, se kterým se pracuje.

Následující příklad cmdletu s parametrem class vrátí instanci WMI třídy, která obsahuje informace o operačním systému:

```
Get-WmiObject –Class Win32_OperatingSystem
```

V oddílu 2.1 bylo čerpáno z [13].

2.2 Objektová roura

Roura je logické spojení mezi dvěma nebo více příkazy. Tedy výstup jednoho příkazu je poslán na vstup příkazu dalšího. Jelikož je PowerShell objektově založený, tak i roura v tomto prostředí pracuje s objekty a nikoliv textem, jak je tomu u ostatních shellů. Ke spojení příkazů se využívá symbol „|“.

Využití roury lze ukázat na tomto příkladu:

```
Get-WmiObject –Class Win32_Process | Select-Object Name |Sort-Object Name
```

Nejprve je získána instance třídy, která obsahuje informace o běžících procesech. Tato třída zahrnuje mnoho vlastností, kvůli kterým by výpis procesů byl příliš dlouhý a nepřehledný. Dalším krokem je proto odfiltrování všech vlastností kromě názvů procesů pomocí cmdletu Select-Object. Pro lepší přehlednost je ještě použit cmdlet Sort-Object, který zajistí seřazení všech procesů podle zadaného parametru. Výsledek tedy bude výpis názvů všech běžících procesů seřazených sestupně podle názvu.

V oddílu 2.2 bylo čerpáno z [14].

3 Praktické využití WMI při spravování sítě

V této kapitole budou uvedeny praktické ukázky, jak lze pomocí WMI spravovat a sledovat počítače v síti. Skripty budou napsány v jazyce PowerShell 2.0 a otestovány v operačním systému Microsoft Windows 7. Některé z těchto postupů budou dále využity při řešení praktické části této práce. WMI obsahuje velké množství dostupných tříd, takže není problém získat detailní informace o jakékoliv části lokálního nebo vzdáleného počítače. Toto se hodí ve správě počítačů v síti, kde nejvíce času zabere napsat a otestovat samotný skript. Přesto se však jedná o mnohem efektivnější řešení než nastavovat každý počítač zvlášť, protože jednou napsaný skript lze dále využít bez omezení počtu spuštění.

3.1 Inventarizace počítačů

Inventarizaci počítačů lze díky WMI a PowerShellu plně automatizovat. Následující demonstrační funkce nejprve načte seznam počítačů z textového souboru. Následně získá pomocí WMI tříd Win32_ComputerSystem, Win32_Processor a Win32OperatingSystem potřebné informace a všechna data vyexportuje do Excelu, kde je lze dále zpracovávat.

```
function ZiskejInformace{

      #Deklarace parametru funkce
      Param([string]$Cesta)

      #Vytvoření nového Excelového souboru
      $Excel = New-Object -ComObject Excel.Application
      $Excel.visible = $True
      $Excel = $Excel.Workbooks.Add()

      #Přiřazení sešitu dané proměnné
      $Sesit1 = $Excel.Worksheets.Item(1)

      #Vytvoření popisků
      $Sesit1.Cells.Item(1,1) = "Název počítače"
      $Sesit1.Cells.Item(1,2) = "Operační systém"
      $Sesit1.Cells.Item(1,3) = "Typ procesoru"
      $Sesit1.Cells.Item(1,4) = "Počet logických jader"
      $Sesit1.Cells.Item(1,5) = "Paměť RAM (GB)"

      #Pomocné proměnné
      $JmenaPocitacu = Get-Content -Path $Cesta
      $CisloRady = 2

      #Postupné procházení všech počítačů v seznamu
      foreach($Pocitac in $JmenaPocitacu)
         {
            #Test dostupnosti daného počítače
            if(Test-Connection($Pocitac) -Count 1 -Quiet){


            #Uložení potřebných WMI tříd do proměnných
            $Informace1 = Get-WmiObject -Class Win32_ComputerSystem `
            -ComputerName $Pocitac
            $Informace2 = Get-WmiObject -Class Win32_Processor `
            -ComputerName $Pocitac
            $Informace3 = Get-WmiObject -Class Win32_OperatingSystem `
```

```
            -ComputerName $Pocitac

            #Vyplnění patřičných buněk požadovanými informacemi
            $Sesit1.Cells.Item($CisloRady, 1) = $Informace1.Name
            $Sesit1.Cells.Item($CisloRady, 2) = $Informace3.Caption
            $Sesit1.Cells.Item($CisloRady, 3) = $Informace2.Name
            $Sesit1.Cells.Item($CisloRady, 4) =
            $Informace2.NumberOfLogicalProcessors
            $Sesit1.Cells.Item($CisloRady, 5) =
            [math]::Round(($Informace1.TotalPhysicalMemory/1GB),2)

            }else{
            #Popisek použitý v případě nedostupnosti počítače
            $Sesit1.Cells.Item($CisloRady, 1) = "Počítač není dostupný!"
            }
            #Zvýšení hodnoty proměnné o jedna
            $CisloRady++
        }
}#Konec funkce
```

Funkce má jako parametr cestu k textovému souboru (například C:\Seznam.txt), který obsahuje seznam s počítači a je volána následovně:

```
ZiskejInformace{cesta}
```

U hardware však možnosti WM nekončí a pokud by bylo třeba získat seznam nainstalovaného softwaru, lze využít třídu Win32_Product.

3.2 Konfigurace síťových adaptérů

WMI obsahuje několik tříd, díky kterým lze pracovat se síťovými adaptéry. Jako první ukázka poslouží zobrazení seznamu všech fyzických a virtuálních síťových adaptérů. K tomu bude použita WMI třída Win32_NetworkAdapter:

```
Get_WmiObject –Class Win32_NetworkAdapter | select DeviceID, Name
```

Výsledkem bude požadovaný seznam, který obsahuje názvy a identifikační čísla těchto adaptérů. S touto třídou je dále možné zakazovat a povolovat dané adaptéry. V další ukázce je využita metoda Disable, která zakáže adaptér s identifikačním číslem 7:

```
Get-WmiObject -Class Win32_NetworkAdapter -Filter DeviceID=7 | Invoke-
WmiMethod -Name Disable
```

Ke konfiguraci slouží třída Win32_NetworkAdapterConfiguration. Tato třída reprezentuje atributy a chování síťového adaptéru. Obsahuje vlastnosti a atributy, které podporují správu protokolů TCP/IP a IPX, jež jsou nezávislé na síťovém adaptéru. Následující funkce demonstruje využití tří metod této třídy při nastavení IP adresy, výchozí brány a DNS serveru daného adaptéru na vzdáleném počítači:

```
function NastavIP{
$Adapter= Get-WmiObject -Class Win32_NetworkAdapterConfiguration
-ComputerName NazevPocitace -Filter Index=7
$Adapter.EnableStatic("192.168.0.1","255.255.255.0")
```

```
$Adapter.SetGateways("192.168.0.1", 1)
$Adapter.SetDNSServerSearchOrder("10.0.0.100")
}
```

Pokud je místo statických adres potřeba použít DHCP, poslouží metoda EnableDHCP:

```
Get-WmiObject -Class Win32_NetworkAdapterConfiguration -ComputerName
NazevPocitace -Filter Index=7 | Invoke-WmiMethod -Name EnableDHCP
```

Další užitečná třída je například Win32_PingStatus, která představuje hodnoty vrácené standardním příkazem ping. Třída Win32_IP4RouteTable zase dokáže zobrazit informace o směrování IP ze systému.

3.3 Správa uživatelů a skupin

Práci s uživatelskými účty má na starosti WMI třída Win32_UserAccount. WMI dostupné ve Windows 7 nepodporuje vytváření a mazání uživatelských účtů. Lze však například zobrazit seznam všech účtů na daném počítači:

```
Get-WmiObject -Class Win32_UserAccount -ComputerName NazevPocitace |
select Name
```

Dále se může hodit zobrazení všech momentálně přihlášených uživatelů. WMI sice neumožňuje přímé zjištění toho, který uživatel je přihlášený, ale udržuje spojení mezi třídami pomocí asociačních tříd. Následující funkce získá díky WMI třídě Win32_LoggedOnUser instance všech uživatelských účtů, které jsou momentálně přihlášené do systému. Každá instance asociuje uživatelský účet a relaci přihlášení. Jelikož vlastnost Antecedent obsahuje kromě názvů přihlášených uživatelských účtů ještě například název jmenného prostoru a další, je nutné tyto nepotřebné věci odfiltrovat. Tato ukázka tedy zobrazí názvy přihlášených účtů:

```
function PrihlaseniUzivatele{

#Deklarace parametru funkce
param ([string]$NazevPocitace)

#Ziskání potřebnych dat pomoci WMI tridy Win32_LoggedOnUser
Get-WmiObject -Class Win32_LoggedOnUser -ComputerName $NazevPocitace |
foreach {

      #Vlastnost Antecedent obsahuje uložené účty
      $ucet = $_.Antecedent -split ","

      #Vytvoreni nového objektu s požadovaným údajem
      $uzivatel = New-Object -TypeName PSObject -Property @{

            #Odfiltrování nepotřebných řetězců
            User = ($ucet[1] -split "=")[1] -replace '"', ''
      }

      #Přejmenovani objektu na výstupu
      $uzivatel.PSTypeNames[0] = "PrihlaseniUzivatele"
      $uzivatel
```

```
      }
}
```

Pro práci s uživatelskými skupinami lze využít WMI třídu Win32_Group. Skupiny jsou velmi důležité, protože bez nich by bylo nutné spravovat každý účet zvlášť, což by značně zatížilo odpovědné osoby. Zobrazení všech skupin v daném počítači je následovné:

```
Get-WmiObject –Class Win32_Group –ComputerName NazevPocitace
```

Dále je dobré znát členství uživatelů v jednotlivých lokálních skupinách. Toto platí zejména pro skupinu Administrators, jejíž členové mají veškerá oprávnění. Pro získání potřebných informací je nutné nejprve pomocí třídy Win32_Group získat informace o dané skupině a následně s využitím třídy Win32_UserAccount vypsat všechny uživatelské účty obsažené v této skupině. Možné řešení tohoto problému:

```
function ClenstviUzivatelu{

      #Deklarace parametru funkce
      param ([string]$NazevPocitace)

      #Ziskani informaci o skupinach
      Get-WmiObject -Class Win32_Group -ComputerName $NazevPocitace |
      foreach {

            #Pomocné proměnné
            $skupina = $_.Name
            $domena = $_.Domain

            #WQL dotaz, který zjistí příslušnost daných uživatelů ke
skupinám
$dotaz = "ASSOCIATORS OF {Win32_Group.Domain='$domena',
   Name='$skupina'} WHERE ResultClass = Win32_UserAccount"

            #Využití předchozího dotazu
            Get-WmiObject -query $dotaz -ComputerName $NazevPocitace|
            foreach {

                  #Vytvoření nového objektu s požadovanými údaji
                  $clen = New-Object -TypeName PSObject -Property @{
                  GroupName = $skupina
                  GroupDomain = $domena
                  UserName = $_.Name
                  UserDomain = $_.Domain
                  }

                  #Výpis informací
                  $clen
            }
      }
}
```

3.4 Práce se službami a procesy

Řízení služeb a procesů je důležitá činnost při spravování počítačové sítě. Ať už se jedná o zjištění uživatele, který spustil proces, jenž zabírá příliš mnoho systémových prostředků

nebo pouhé zapnutí určité služby. K řízení služeb bude využita WMI třída Win32_Service. PowerShell sice obsahuje několik cmdletů umožňující správu služeb, ale WMI používá mírně odlišný přístup k práci se službami. Výpis všech dostupných služeb na daném počítači, způsobu jejich spuštění a jeich momentální stav lze vypsat následovně:

```
Get-WmiObject –Class Win32_Service –ComputerName NazevPocitace | select
Name, StartMode, State
```

WMI dále umožňuje vytváření, mazání, vypínání a spouštění služeb. Příklad demonstrující rozdíl mezi vypnutím služby pomocí WMI a cmdletu dostupném v PowerShell, který tuto funkci zastává:

```
#Pomocí PowerShellu
Stop-Service –Name BITS

#Pomocí WMI
Get-WmiObject –Class Win32_Service –Filter „name='BITS'" | Invoke-
WmiMethod –Name StopService
```

Jak je vidět PowerShell v tomto případě nabízí mnohem jednodušší řešení daného problému. Kromě služeb lze podobně pracovat také s procesy, jejichž správu obstarává WMI třída Win32_Process. Zobrazení jejich seznamu spolu s vlastníkem daného procesu a domény, ke které daný proces patří:

```
function VlastnikProcesu {

      #Deklarace parametru funkce
      param ([string]$NazevPocitace)

      #Získání informací o procesech a jejich následný výpis
      Get-WmiObject -Class Win32_Process -ComputerName $NazevPocitace |
      select Name,
      @{Name="Doména";Expression={($_.GetOwner()).Domain}},
      @{Name="Uživatel";Expression={($_.GetOwner()).User}}
}
```

Zastavení určitého procesu na vzdáleném počítači lze provést následovně:

```
Get-WmiObject -Class Win32_Process -ComputerName NazevPocitace -Filter
"Name='NazevProcesu'" | Remove-WmiObject
```

3.5 Manipulace se soubory

PowerShell spolu s WMI poskytuje prostředky pro správu souborů. Lze je mazat, prozkoumávat a manipulovat s nimi, avšak nelze vytvářet nové soubory nebo složky. Při práci se soubory se také místo WMI Win32 třídy využívá třída CIM_DataFile. Nevýhodu může představovat větší časová náročnost při práci se soubory na rozsáhlejším vzdáleném systému, tuto nevýhodu však vyvažuje jednoduchost řešení. Následující příklad demonstruje zobrazení obsahu složky na vzdáleném počítači:

```
$dotaz = "SELECT * FROM CIM_DATAFILE WHERE Drive='C:' AND
Path='\\Cesta\\'"
Get-WmiObject -Query $dotaz -ComputerName NazevPocitace | select Name
```

Pro smazání určitého souboru je postup velice podobný:

```
Get-WMIObject -query "Select * From CIM_DataFile Where name='c:\\Cesta" -
computer NazevPocitace | Remove-WmiObject
```

V kapitole 3 bylo čerpáno z [5], [15], [16].

Závěr

V této práci jsem objasnil princip funkčnosti technologie WMI. Bylo zmíněno i několik standardů, které s WMI souvisejí. Vybral jsem a stručně popsal několik alternativních technologií pro správu sítě, které lze místo WMI využít. Dále jsem se seznámil s nástrojem a programovacím jazykem PowerShell, který byl následně využit jako prostředek pro práci s WMI v poslední kapitole této práce. Byly popsány základní možnosti využití WMI ve správě sítě a nastíněna problematika, týkající se jednotlivých úkonů.

V praktické části bylo cílem vytvořit demonstrační nástroj, který by využíval WMI jako prostředek pro získávání informací o počítačích. Jako ideální programovací jazyk se pro tento účel jevil výše zmíněný PowerShell, který je od základu navržen zejména ke správě systému Microsoft Windows. PowerShell obsahuje zabudované příkazy, které velmi usnadňují práci s WMI. Dále bylo díky spojení s .NET platformou možno vytvořit formulářovou aplikaci, která plní požadovaný účel. Vytvořený program umožňuje zobrazit informace o vybraném vzdáleném počítači.

Literatura

[1] RUSSINOVICH, Mark E, David A SOLOMON a Alex IONESCU. *Windows internals*. 6. vyd. Redmond, Wash.: Microsoft Press, 2012. ISBN 07-356-4873-5.

[2] WMI Architecture [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://msdn.microsoft.com/en-us/library/windows/desktop/aa394553%28v=vs.85%29.aspx

[3] Zprostředkovatel rozhraní WMI protokolu SNMP - přehled [online]. [cit. 2013-05-11]. Dostupné z: http://technet.microsoft.com/cs-cz/library/cc770487%28v=ws.10%29.aspx

[4] COOPERSTEIN, Jeffrey. Windows Management Instrumentation: Administering Windows and Applications Across Your Enterprise. *MSDN magazine* [online]. San Francisco, CA: CMP Media Inc., 2000 [cit. 2013-05-11]. Dostupné z: http://msdn.microsoft.com/cs-cz/magazine/cc302340%28en-us%29.aspx

[5] SIDDAWAY, Richard. *Powershell and WMI*. Shelter Island: Manning, 2012, 514 s. ISBN 16-172-9011-4.

[6] MEYLER, Kerrie. *System center 2012 configuration manager unleashed*. Indianapolis, Ind.: Sams, 2013, 1329 s. ISBN 978-067-2334-375.

[7] Autorizace uživatelů služby WMI a nastavení oprávnění [online]. [cit. 2013-05-11]. Dostupné z: http://technet.microsoft.com/cs-cz/library/cc787533%28v=ws.10%29.aspx

[8] BRAGG, Roberta. Securing Remote Management with WMI. *Redmond Magazine* [online]. 2002 [cit. 2013-05-11]. Dostupné z: http://redmondmag.com/articles/2002/02/01/securing-remote-management-with-wmi.aspx

[9] Securing a Remote WMI Connection [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://msdn.microsoft.com/en-us/library/windows/desktop/aa393266%28v=vs.85%29.aspx

[10] Introduction to Cisco IOS NetFlow - A Technical Overview [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html

[11] Common Information Model [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://msdn.microsoft.com/cs-cz/library/windows/desktop/aa389234%28v=vs.85%29.aspx

[12] Querying with WQL [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://msdn.microsoft.com/en-us/library/windows/desktop/aa392902%28v=vs.85%29.aspx

[13] KOPCZYNSKI, Tyson. *Windows PowerShell unleashed*. Indianapolis, Ind.: Sams, 2007, 306. ISBN 06-723-2953-0.

[14] CHWICHTENBERG, Holger. *Essential PowerShell*. Upper Saddle River, N.J.: Addison-Wesley, 2008, 478 s. ISBN 06-723-2966-2.

[15] PAYETTE, Bruce. *Windows PowerShell in action*. 2. vyd. Shelter Island, N.Y.: Manning, 2011, 984 s. ISBN 19-351-8213-7.

[16] JONES, Don. *Learn Windows PowerShell in a month of lunches*. Shelter Island, NY: Manning Publications, 2011, 309 s. ISBN 16-172-9021-1.

Příloha A – Instalační příručka

Pro spuštění programu je nutné mít nainstalovaný Windows PowerShell ve verzi 2.0 nebo vyšší. PowerShell 2.0 je dostupný pro operační systémy Windows XP, Windows Server 2003, Windows Vista, Windows 7 a Windows Server 2008 R2, přičemž na posledních dvou je od základu nainstalován. Windows Management Framework Core package, který mimo PowerShellu 2.0 obsahuje také Windows Remote Management 2.0 lze stáhnout z adresy http://support.microsoft.com/kb/968930/en-us.

PowerShell dále vyžaduje .NET Framework alespoň ve verzi 2.0. Nejnovější dostupná verze (.NET Framework 4.5) je dostupná ke stažení z adresy http://www.microsoft.com/en-in/download/details.aspx?id=30653.

PowerShell ve výchozím nastavení obsahuje zabezpečení, které zabrání spuštění všech skriptů. Před startem programu je tedy nutné toto nastavení změnit. Na výběr jsou následující možnosti:

- **Restricted** – Žádné skripty nemohou být spouštěné.
- **AllSigned** – Mohou být spuštěny pouze skripty podepsané od důvěryhodného vydavatele.
- **RemoteSigned** – Nelze spouštět vzdáleně uložené skripty.
- **Unrestricted** - Mohou být spouštěné jakékoliv skripty.

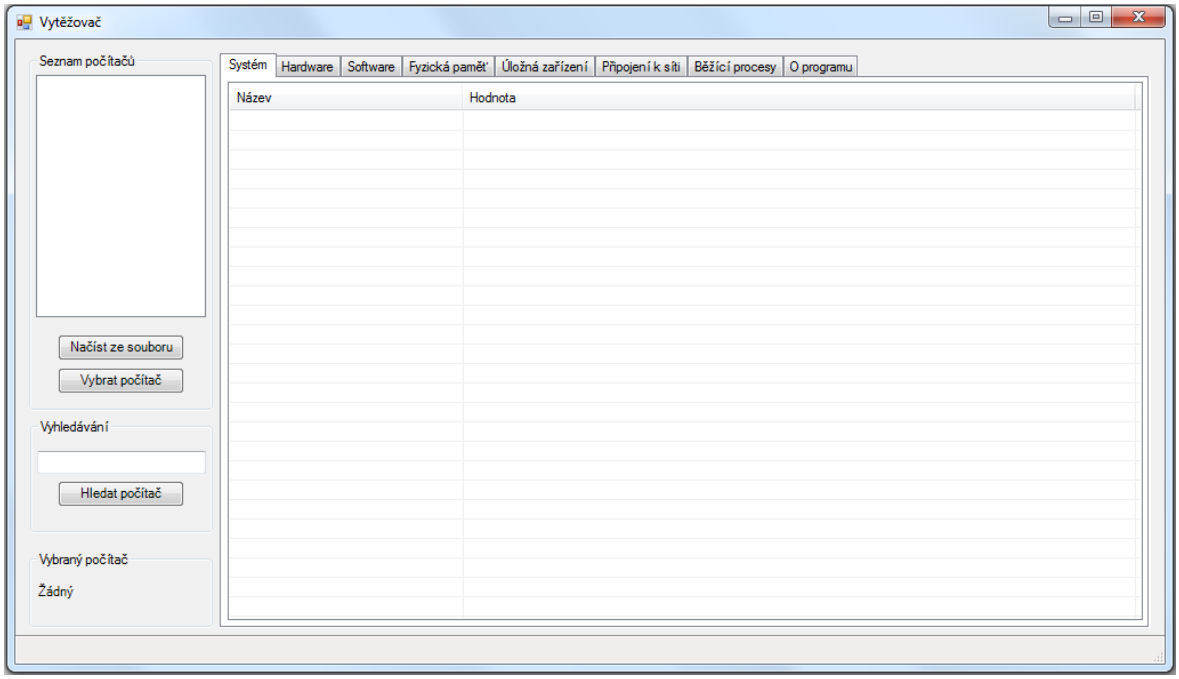
Následující cmdlet demonstruje změnu bezpečnostní politiky PowerShellu:

```
Set-ExecutionPolicy unrestricted
```

Příloha B – Uživatelská příručka

Obsluha programu je velice snadná. Před samotným zobrazením informací o daném počítači je nutné zadat jeho název. Toho lze docílit dvěma způsoby. První možností je načtení textového souboru, ve kterém jsou uložené názvy počítačů tak, aby na každém řádku bylo pouze jeden, přičemž je není třeba nijak oddělovat. Poté je třeba vybrat jméno ze zobrazeného seznamu a kliknout na tlačítko "Vybrat počítač". Program ověří dostupnost dané stanice a v případě, že bylo spojení úspěšně navázáno o něm zobrazí informace. Druhá možnost spočívá v zadání názvu zvoleného počítače do pole pod popisek "Vyhledávání". Dále je průběh stejný jako v předchozím případě.

Samotné získání informací o daném počítači může trvat déle (až v řádech desítek vteřin). Některé informace se nemusí zobrazit v případě, že daný počítač neobsahuje danou komponentu. Pokud program počítač nenalezne nebo zjistí, že je nedostupný, oznámí tuto skutečnost v levé části spodní lišty. Na obrázku 3 je znázorněno rozložení jednotlivých prvků programu.



Obrázek 3 - Demonstrační program praktické části bakalářské práce