VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

WEBOVÝ NÁSTROJ PRO SLEDOVÁNÍ STAVU SÍŤOVÝCH TISKÁREN

WEB TOOL FOR MONITORING OF THE STATUS OF NETWORK PRINTERS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **RADIM LIPKA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. RUDOLF ČEJKA**
SUPERVISOR

BRNO 2017

Vysoké učení technické v Brně - Fakulta informačních technologií

Centrum výpočetní techniky Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Lipka Radim**
Obor: Informační technologie
Téma: **Webový nástroj pro sledování stavu síťových tiskáren**
Web Tool for Monitoring of the Status of Network Printers
Kategorie: Počítačové sítě

Pokyny:

1. Prostudujte existující systémy pro sledování stavu síťových tiskáren.
2. Zjistěte, jakými způsoby lze získávat informace z tiskáren různých výrobců.
3. Navrhněte aplikaci, která bude sledovat, ukládat a zobrazovat provozní stav vybraných tiskáren.
4. Implementujte a zkušebně nasaďte aplikaci v rámci fakultní sítě.
5. Zhodnoťte dosažené výsledky a možnosti dalšího vývoje.

Literatura:

- Barevný tisk v síti pod kontrolou. IT Systems. 2009, č. 7-8. ISSN 1802-002X. Dostupné z: https://www.hwforum.cz/tiskarny/barevny-tisk-v-siti-pod-kontrolou.html
- Monitoring tiskáren: Mějte přehled o veškerém tisku. BusinessIT. 2012. ISSN 1805-0522. Dostupné z: http://www.businessit.cz/cz/monitoring-tiskaren-prehled-o-tisku-software-download-zdarma-uziti.php
- BERGMAN, R. et al. Printer MIB v2. IETF RFC 3805, 2004. Dostupné z: https://tools.ietf.org/html/rfc3805

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2 zadání

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Čejka Rudolf, Ing.**, CVT FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Tato práce se zabývá tvorbou nástroje, který průběžně monitoruje stav síťových tiskáren a jejich spotřebních materiálů. Tento nástroj slouží správcům sítě k udržování tiskáren v provozuschopném stavu. Důraz je kladen na proces získávání a ukládání dat tiskáren různých výrobců. Data jsou poté přehledně zobrazována webovým rozhraním za využití frameworku Flask. V teoretické části práce analyzuje dostupná řešení na internetu a navrhuje nový přístup získávání dat. Následuje návrh, který zahrnuje mnohá vylepšení a implementace oddělené frontendové a backendové části, jenž jsou obě implementovány v jazyce Python. Závěrem je aplikace nasazena na fakultní síti FIT VUT a jsou navrhnuta další možná vylepšení.

Abstract

This thesis describes the process of creating a tool for continuous monitoring of network printers and their consumables. This tool is used by network administrators to maintain printers in operational state. The major task is collecting and saving relevant data of printers of different vendors. The data are displayed to the user through a web tool using the framework Flask. Solutions available on the internet are analyzed in the theoretical part of the thesis, along with a draft of a new way to collect data. The thesis also provides information on the design of some improvements and separate implementations of frontend and backend, both implemented in Python. Eventually the application is launched on the FIT BUT network and more possible improvements to be done are mentioned.

Klíčová slova

síťová tiskárna, monitorování, spotřební materiály, SNMP, HTTP

Keywords

network printer, monitoring, consumables, SNMP, HTTP

Citace

LIPKA, Radim. *Webový nástroj pro sledování stavu síťových tiskáren.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Čejka Rudolf.

Webový nástroj pro sledování stavu síťových tiskáren

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Rudolfa Čejky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Radim Lipka
12. května 2017

Poděkování

Rád bych poděkoval panu inženýrovi Rudolfovi Čejkovi za odborné vedení při psaní této bakalářské práce.

Obsah

1 Úvod **3**
- 1.1 Uvedení do kontextu . . . 3
- 1.2 Cíl . . . 3
- 1.3 Struktura práce . . . 3

2 Analýza problematiky **4**
- 2.1 Dostupná řešení na internetu . . . 4
  - 2.1.1 Xerox Device Agent Lite (XDA-Lite) . . . 5
  - 2.1.2 4 Office Network Printer Discovery . . . 7
  - 2.1.3 BRAdmin Professional . . . 9
  - 2.1.4 Shrnutí . . . 11
- 2.2 Monitorovaná data . . . 11
  - 2.2.1 Obecné spotřební materiály . . . 12
  - 2.2.2 Spotřební materiály laserových tiskáren . . . 12
  - 2.2.3 Spotřební materiály inkoustových tiskáren . . . 14
  - 2.2.4 Shrnutí . . . 14
- 2.3 Způsoby získávání dat z tiskáren . . . 14
  - 2.3.1 SNMP . . . 15
  - 2.3.2 HTTP . . . 18
  - 2.3.3 Shrnutí . . . 19

3 Návrh řešení **20**
- 3.1 Specifikace požadavků . . . 20
  - 3.1.1 Základní požadavky . . . 20
  - 3.1.2 Požadavky odvozené . . . 21
- 3.2 Základní návrh aplikace . . . 22
  - 3.2.1 Získávání dat . . . 22
  - 3.2.2 Konfigurace . . . 23
  - 3.2.3 Databáze . . . 24
  - 3.2.4 Opakované získávání dat . . . 25
  - 3.2.5 Zobrazení dat ve frontendu . . . 26

4 Implementace **27**
- 4.1 Výběr programovacího jazyka . . . 27
- 4.2 Implementace backendu . . . 27
  - 4.2.1 Parser argumentů - `ArgumentsParser` . . . 28
  - 4.2.2 Parser konfigurace - `ConfigurationParser` . . . 29
  - 4.2.3 Načítač modulů tiskáren - `ModulesLoader` . . . 30

4.2.4 Databázový manager - `DatabaseManager` . . . 30
4.2.5 Kalendář událostí - `EventsCalendar` . . . 30
4.2.6 Tiskárny - `Printer` . . . 30
4.2.7 Vázací manager - `PrinterBinder` . . . 30
4.2.8 Moduly pro zjednodušení získávání dat - `SnmpGetter, HttpGetter` 31
4.3 Implementace frontendu . . . 31
4.3.1 Parser argumentů - `ArgumentsParser` . . . 32
4.3.2 Parser konfigurace - `ConfigurationParser` . . . 32
4.3.3 Databázový manager - `DatabaseManager` . . . 32
4.3.4 Manager dat - `PrinterManager` . . . 32
4.3.5 Webový server - `flask` . . . 33
4.4 Použité knihovny . . . 33
4.4.1 Databáze - *pymysql* . . . 33
4.4.2 Konfigurace - *lxml* . . . 33
4.4.3 Webový server - *flask* . . . 33
4.4.4 Další utility . . . 34

5 Nasazení v síti **35**
5.1 Postup . . . 35
5.2 Výsledky . . . 35

6 Závěr **37**
6.1 Rozšíření do budoucna . . . 37
6.2 Zhodnocení práce . . . 37

Literatura **39**

Přílohy **41**

A Obsah přiloženého paměťového média **42**

B Souborová struktura projektu aplikace PriMoVi **43**

C Ukázky webového výstupu aplikace **45**

Kapitola 1

Úvod

1.1 Uvedení do kontextu

S rostoucí síťovou infrastrukturou podniků přibývá zařízení, která napomáhají ulehčovat práci, zajišťovat provoz sítě, či provádět úkony nezbytné k provádění práce (scannery, tiskárny, kamery, routery, ...). Tato zařízení jsou připojena do sítě a umožňují tak s okolním světem komunikovat. Většina z nich funguje po prvním nakonfigurování samostatně a není potřeba se o ně dále starat. Všechna zařízení však potřebují být udržována v provozním stavu, což je nezbytné průběžně vykonávat, aby dál plnily svou funkci. Velice neefektivní by bylo, kdyby zařízení musel každý den obcházet údržbář, pokud máme možnost se zařízením po síti komunikovat a zjišťovat tak jeho stav. Nabízí se nám tedy možnost zefektivnit zajištění provozuschopnosti zařízení a plánování oprav, nebo výměn se zařízeními spojených, aniž by k zařízením musel kdokoliv fyzicky dojít.

1.2 Cíl

Cílem práce je navrhnout a implementovat aplikaci, která monitoruje stav tiskáren různých výrobců a ulehčuje tak práci správcům sítí s udržováním tiskáren v provozuschopném stavu a jejich monitorováním. Práce se zaměřuje především na získávání, zpracování a ukládání dat tiskáren různých výrobců.

1.3 Struktura práce

Kapitola 2 se zaměřuje na hotová řešení, která můžeme najít na internetu. Popisuje, jaká užitečná data můžeme z tiskáren získávat a jakým způsobem. Další kapitola 3 je věnována specifikaci požadavků a návrhu výsledného řešení. Následující kapitola 4 se zabývá detailním popisem a implementací dílčích součástí. Na závěr, v kapitole 5, je aplikace nasazena na lokální síti FIT VUT a porovnána s konkurenčními řešeními.

Kapitola 2

Analýza problematiky

Na úvod analyzujeme dostupná řešení jiných výrobců, zjistíme, jaký spotřební materiál tiskáren lze monitorovat a navrhneme metody získávání dat z tiskáren.

2.1 Dostupná řešení na internetu

Nejprve analyzujeme již existující aplikace pro monitorování stavu síťových tiskáren a získané vědomosti použijeme při návrhu naší aplikace. Analýzou se poučíme z chyb a inspirujeme přednostmi jiných řešení. Hlavním cílem analýzy aplikací jsou (seřazeno prioritně):

- principy získávání dat z tiskáren,
- podporovaná zařízení,
- funkce aplikace,
- zobrazované informace,
- obecné informace,
- prerekvizity.

Jelikož při testování aplikací nelze obsáhnout veškeré tiskárny různých výrobců, byly aplikace testovány na modelech, nacházejících se ve fakultní síti FIT VUT. Za každého výrobce byl vybrán jeden model.

Přiřazení modelů je následující:[1]

Výrobce	Model
Konica Minolta	Bizhub C3110
Xerox	Phaser 6300DN
Brother	MFC-9970CDW
HP	Color LaserJet CM2320 fxi MFP
Samsung	CLX-6220 Series
Canon	MF4360-4390
OKI	Color MC561dn
Epson	WF-7610 Series

Tabulka 2.1: Testované typy tiskáren

2.1.1 Xerox Device Agent Lite (XDA-Lite)

Výrobce: Xerox [14, 13]
Verze: 5.3.5
Typ: desktopová aplikace
OS: Windows XP a výše, Windows Server 2003 a výše
Licence: zdarma

Princip získávání dat

Aplikace získává data pomocí SNMP protokolu (viz 2.3.1) verze 1. Pro správnou funkčnost SNMPv1 lze zadat *community string*, který je implicitně nastaven na hdnotu *public*. Aplikace získává data z těchto MIB databází (viz 2.3.1): *Printer MIB v1/v2*, *Printer Finishing MIB*, *Host resources MIB v1/v2*, *MIB II for TCP Based internet*. Získávány jsou následující informace: identifikace tiskárny (sériové číslo, model), vlastnosti tiskárny (podavače, nádobky), stav tiskárny, spotřeba (inkoust, tonery), podporované tiskové protokoly, komunikační protokoly a mnohé další.

Podporované tiskárny

Aplikace je primárně určená pro tiskárny firmy Xerox. U tiskáren jiných výrobců negarantuje podporu. Dalšími podporovanými tiskárnami s omezenými funkčnostmi při testování byly: Konica Minolta, Brother, HP, Samsung, Epson, Canon a OKI.

Funkce aplikace

Přidávání tiskáren lze provádět automaticky ze zadané podsítě, pomocí IP adresy, nebo DNS jména. Pro přidávání tiskáren lze rovněž využít importu databáze, nebo lze konfiguraci exportovat. Z tiskáren jsou průběžně získávány informace, které se ukládají do lokální databáze. Pokud na tiskárně dojde k chybě, je uživatel buď notifikován vyskakovacím oknem aplikace, nebo varováním v podobě e-mailové zprávy. Jedná se o varování týkající se kritických chyb, potřeby výměny či doplnění spotřebních materiálů, nebo chyb aplikace.

[1] Kdykoliv bude v textu zmíněn výrobce tiskárny (nebude-li uvedeno jinak), je tím myšlen model, který byl použit při testování.

Zobrazované informace

Pro zobrazení dat existují v aplikaci dva módy. První je v podobě dlaždic, kde lze vidět přehled aktuálního stavu (viz obrázek 2.1 níže). Druhý mód zobrazuje aktuální seznam, který lze modifikovat filtrováním nebo vybráním položek k zobrazení. V obou těchto zobrazeních je možné využít filtru, který umožňuje zobrazení tiskáren podle daných kritérií. Po rozkliknutí dané tiskárny máme k dispozici podrobnější aktuální popis. Ten je dělen do kategorií: status, spotřeby, informace, vlastnosti, měření využití, souhrn spotřeby a detaily spotřeby. Nás zajímají především údaje týkající se spotřebních materiálů a provozních stavů tiskárny. Pro testované typy tiskáren aplikace zobrazuje:

Tiskárna	Spotřeba toneru	Status papíru	Počet stran	Run time	Status
Xerox	procenta	slovně	ano	ano	ano
Konica Minolta	procenta	neuveden	sno	ano	ano
Brother	slovní popis	slovně	ano	ano	ano
Samsung	procenta	slovně	ano	ano	ano
HP	procenta	slovně	ano	ano	ano
Epson	procenta	neuveden	ano	ano	ano
Canon	nedostupné	slovně	ano	ano	ano
OKI	procenta	neuveden	ano	ano	ano

Tabulka 2.2: Výsledky testování tiskáren v aplikaci XDA-lite

Prerekvizity aplikace

Aplikace vyžaduje internetové připojení a podporu protokolu IPv4, nebo IPv6. Taktéž je nutno mít povoleno směrování SNMP protokolu v síti. Nemůže být nainstalována vedle již stávajícího řešení aplikací rodiny XDA. Aplikace ke svému běhu vyžaduje *xSQL Server 2005 Compact Edition*, pomocí něhož si do databáze ukládá data o tiskárnách a své konfiguraci.

Shrnutí

Aplikace získává data pouze pomocí protokolu SNMP. U některých tiskáren není schopna získat všechna potřebná data spotřebních materiálů. Umožňuje ukládání pouze jednorázových hodnot, nemožných sledovat zpětně v čase. Rozhraní aplikace je velice povedené, zvláště přínosné jsou funkce filtrování a řazení. Umožňuje rovněž zasílání informací o stavech správcům pomocí e-mailové služby.

Obrázek 2.1: XDA-Lite: Hlavní nabídka - dlaždicový přehled tiskáren

2.1.2 4 Office Network Printer Discovery

Výrobce: 4 Office
Verze: 1.0
Typ: desktopová aplikace
OS: Windows XP a výše
Licence: zdarma

Princip získávání dat

Aplikace získává data pomocí SNMP protokolu verze 1, neumožňuje ovšem měnit hodnotu *community stringu*, který je napevno nastaven na hodnotu *public*. Zjištění dalších metod nebylo z důvodu chybějící dokumentace možné. Po zachytávání paketů pomocí aplikace *wireshark* jsem zjistil, že aplikace k získávání používá pouze SNMP.

Podporované tiskárny

Aplikace zvládla rozpoznat veškeré testované tiskárny, jenž SNMP podporují. Jedná se o: Konica Minolta, Brother, Epson, HP, Xerox, Samsing, Canon a OKI.

Funkce aplikace

Aplikace nabízí automatické skenování sítě, ve které se nacházíme, případně zadávání tiskáren ručně zadáváním IP adres. Získané výsledky lze exportovat do CSV souboru. Tím však veškerá funkčnost končí. Pokud chceme po příštím startu opět získat data, je potřeba znovu zadávat IP adresy tiskáren, nebo skenovat síť.

Zobrazované informace

K zobrazení dat slouží obyčejná tabulka. Ta zobrazuje IP adresu, status, popis zařízení, zprávu, sériové číslo, počet vytištěných stran, počet barevných vytištěných stran, stav pouze černého toneru a čas běhu celkem. Pro testované typy tiskáren aplikace zobrazuje:

Tiskárna	Spotřeba černého toneru	Status papíru	Počet stran	Run time	Status
Xerox	procenta	ne	ano	ano	ano
Konica Minolta	nedostupné	ne	ano	ano	ne
Brother	nedostupné	ne	ano	ano	ano
Samsung	procenta	ne	ano	ano	ne
HP	procenta	ne	ano	ano	ano
Canon	nedostupné	ne	ano	ano	ano
OKI	procenta	ne	ano	ano	ano
Epson	procenta	ne	ano	ano	ne

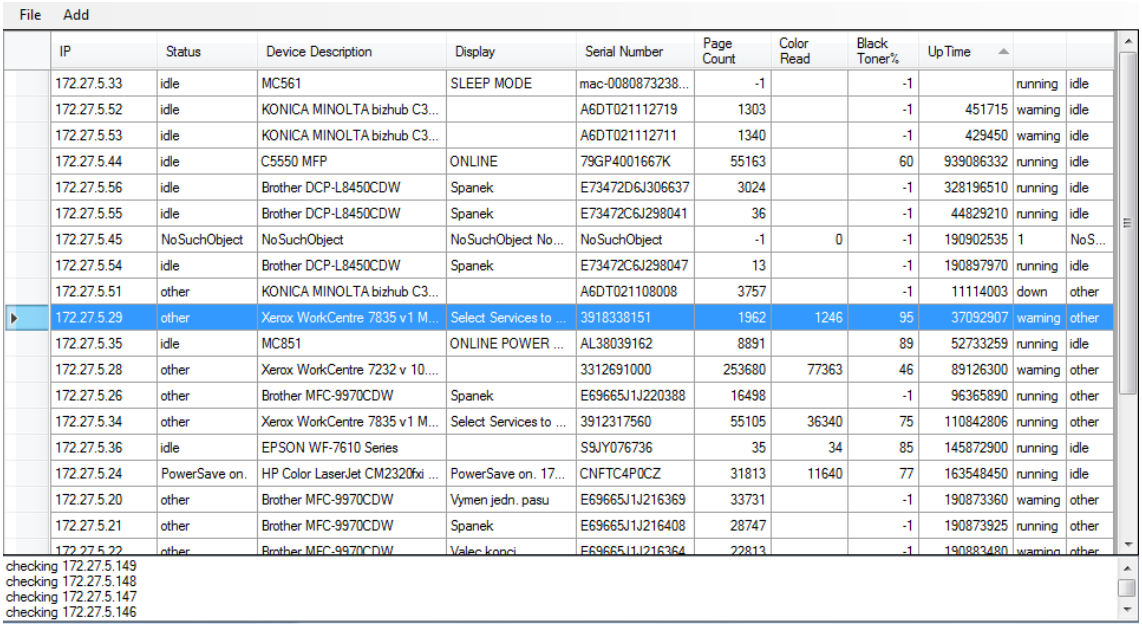
Tabulka 2.3: Výsledky testování tiskáren v aplikaci 4 Office Network Printer Discovery

Prerekvizity aplikace

Aplikaci lze stáhnout jako spustitelný soubor. Ke svému běhu potřebuje OS Windows, připojení k síti a povolení směrování protokolu SNMP.

Shrnutí

Aplikace umožňuje zjišťování informací pouze pomocí SNMP. Kromě černého toneru nezobrazuje žádné další spotřební materiály. Získané informace si neukládá do časové osy. Zobrazování informací je velice nevkusné a neumožňuje ani filtrování či skupinování. Aplikace si po příštím spuštění nepamatuje svou původní konfiguraci.



Obrázek 2.2: Network Printer Discovery: Hlavní nabídka - přehled tiskáren

2.1.3 BRAdmin Professional

Výrobce: Brother [1]
Verze: 3.50.0002
Typ: desktopová aplikace
OS: Windows XP a výše
Licence: zdarma

Princip získávání dat

Aplikace získává data pomocí SNMP protokolu verze 1. Používat lze i SNMP protokol verze 3. V případě protokolu v3 je nutno zadat další nezbytné údaje nutné k autentizaci a šifrování komunikace.

Podporované tiskárny

Primární zaměření aplikace je na tiskárny společnosti Brother. Aplikace zvládla rozpoznat další testovaná zařízení. Jedná se o tiskárny: HP, Samsung, Epson, OKI a Xerox.

Funkce aplikace

Aplikace umožňuje průběžné skenování sítě a zjišťování stavu tiskáren. Tiskárny lze přidávat ručně zadáním IP adresy, DNS jména, rozsahu podsítě, nebo importováním konfigurace ve vstupním souboru (formát *adresa/jméno*). Dále umožňuje konfiguraci některých jejich atributů, jako například jméno správce, nebo umístění skrz SNMPv3. V případě chyby, nebo varování na tiskárnách umožňuje aplikace poslat e-mail správci. Dále je možno aktualizovat firmware.

Zobrazované informace

Zobrazování informací můžeme rozdělit do dvou částí. První z nich je zobrazení obecných informací (jméno, umístění, stav, adresa, jméno správce, ... (viz obrázek 2.3 níže)) ve sloupcích tabulky pro dané tiskárny, jimž náleží jednotlivé řádky. Zobrazované sloupce lze nastavit v konfiguraci. Program rovněž umožňuje tiskárny filtrovat podle několika atributů: MAC adresa, DNS jméno, IP adresa, jméno modelu, umístění a kontaktní osoby, nikoliv však podle hodnot spotřebních materiálů. Při rozkliknutí dané tiskárny můžeme tiskárnu konfigurovat, či přistoupit na její webový server. Pokud chceme zjistit další informace o tiskárnách, je nutno ji přiřadit do monitorování. Výsledkem toho je druhé zobrazení, jímž je webová stránka s jednou tabulkou. Ta na řadcích obsahuje tiskárny a ve sloupcích jejich argumenty. U tiskáren jiných výrobců vidíme pouze základní údaje, jako počet vytištěných stran a základní informace týkající se jména tiskárny, správce, ... Pro tiskárny společnosti Brother vidíme zbývající životnosti jednotlivých spotřebních materiálů a také počty měnění jednotlivých součástí. Pro testované typy tiskáren aplikace zobrazuje:

Tiskárna	Spotřeba toneru	Status papíru	Počet stran	Run time	Status
Xerox	ne	ne	ano	ne	ano
Brother	ano	ano	ano	ano	ano
Samsung	ne	ne	ano	ne	ne
HP	ne	ne	ano	ne	ano
Epson	ne	ne	ano	ne	ano
OKI	ne	ne	ano	ne	ano

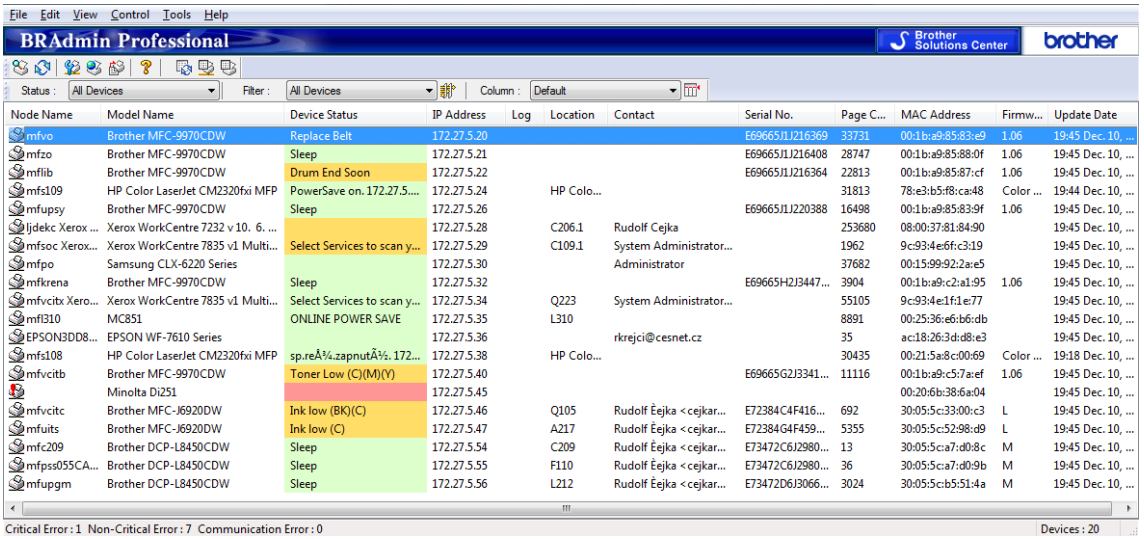
Tabulka 2.4: Výsledky testování tiskáren v aplikaci BRAdmin Professional

Prerekvizity aplikace

Aplikace potřebuje připojení k síti s podporu protokolů TCP/IP, nebo IPX/SPX. Opět je nutno mít povolené směrování protokolu SNMP v síti.

Shrnutí

Aplikace pracuje i s protokolem SNMPv3, kvůli zabezpečení při možnosti konfigurovat zařízení. V logovacích souborech aplikace u tiskáren jiných výrobců zobrazuje jen základní informace, které nelze nijak řadit ani filtrovat.



Obrázek 2.3: BRAdmin professional: Hlavní nabídka - přehled tiskáren

2.1.4 Shrnutí

Ze všech aplikací nyní vyjmenujeme získané poznatky, které se můžou hodit při tvorbě naší aplikace. Jedná se o získávání dat z tiskáren pomocí protokolu SNMP. Každá aplikace však podporovala různý počet MIB databází, což se projevilo v počtu možných zjistitelných informací. Přínosnou funkcí bylo umožnění filtrování dat podle různých parametrů, což může správci sítě umožnit přehlednější zobrazení a provádění efektivnější údržby. Dále pak možnost notifikace správce o kritických stavech a varováních sledovaných zařízení. Aplikacím však některé funkce chyběly. Aplikace neumožňovaly zobrazení stavu spotřebních materiálů v časové ose, což by umožnilo správci sítě mít lepší přehled o vývoji stavů tiskáren v čase. Aplikace rovněž nedávaly možnost uživateli vybrat si, které věci chce sledovat, a které nikoliv. Taktéž pak neumožňovaly přidání vlastních metod, kterými by šlo informace získávat (výrobci aplikací stěží mohou sami pokrýt veškeré typy tiskáren).

2.2 Monitorovaná data

Moderní (dnes skoro veškeré) tiskárny obsahují vestavěný počítač, který se stará o správnou funkčnost celého stroje. Zajišťuje interakci s uživatelem přes displej, interakci v rámci síťového připojení, provádění tiskových úloh a v neposlední řadě provádění monitorování. Aby tiskárna mohla tisknout, musí být veškeré spotřební materiály dostupné. K zjišťování dostupnosti a jejich stavu se používá několik metod:

- zjišťování pomocí čidel, která udávají přesný stav měřené součásti,
- zjišťování pomocí aktuálně vytištěného počtu stran a životnosti dané součásti dané počtem stran, po kterém přestane být použitelná.

Jakmile tiskárna zjistí, že některý ze spotřebních materiálů došel, nebo nastala jiná chyba, musí uživatele informovat, aby chybu opravil (výměnou, doplněním, ...). Tuto notifikaci provádí pomocí displeje, kde vypíše chybovou hlášku. Průběžně lze také sledovat stav všech součástí pomocí webového rozhraní, displeje, nebo vytisknutím stavu na papír. V podkapitole 2.3 si popíšeme, jaké jiné metody se dají použít k zjišťování těchto údajů.

Aby správce sítě mohl průběžně kontrolovat a zajišťovat provozuschopný stav tiskáren, potřebuje vědět, jakými spotřebními materiály tiskárna disponuje. Tyto spotřební materiály se dělí nejprve podle typu tiskárnny. My se budeme zabývat pouze inkoustovými a laserovými. V rámci laserových tiskáren však existuje velké množství různých spotřebních materiálů, které jsou závislé na výrobci a konkrétním typu zařízení. Existují různé variace spotřebních materiálů, jenž lze jen velmi těžko jednotně shrnout. Následující výpis spotřebních materiálů je tedy orientační a je u každé tiskárny jiný. Většina spotřebních materiálů je zjištěna nejen z literatury [2], ale také z výpisů spotřebních materiálů pomocí protokolu SNMP námi sledovaných fakultních tiskáren.

2.2.1 Obecné spotřební materiály

Veškeré tiskárny obsahují vstupní a výstupní zásobník(y) jenž obsahuje(í) média (papír, ...) na které je tisknuto. Tato média je potřeba neustále doplňovat, nejedná se však o práci správce. V případě uvíznutí média však musí správce tiskáren dostat notifikaci a chybu neprodleně odstranit.

Obrázek 2.4: Obecné schéma tiskárny

2.2.2 Spotřební materiály laserových tiskáren

Jedná se o tiskárny, které obsahují velký počet různých spotřebních materiálů, které je potřeba průběžně vyměňovat (doplňovat).

Toner

Je jemný prášek, připomínající až tekutý stav, pomocí kterého se zobrazuje daná bitmapa na papír. V závislosti na barevném modelu a typu tiskárny jich může být v tiskárně několik. Při použití standardního CMYK modelu jsou to čtyři tonery. V případě jednobarevného pouze jeden. V závislosti na typu modelu se dají pořídit (seřazeno dle ceny vzestupně):

- toner v lahvičce, kdy je nutné doplnit cartridge tiskárny pomocí přesypání,
- toner v cartridge, kterou stačí vyměnit za již použitou cartridge v tiskárně,
- toner/developer cartridge obsahující více věcí současně. Jedná se o toner, nádobku (toner hoper) a tonerový válec (developer),
- print cartridge je kombinace předchozího spolu s hlavním fotocitlivým válcem.

Tonerový válec (developer)

Jedná se o válec, který předává nabitý toner hlavnímu fotocitlivému válci. U tiskáren, které neobsahují toner/developer, nebo print cartridge (jenž obsahují mimo jiné i tonerový válec) musí tonerový válec měnit specializovaný odborník, neboť je jeho samostatná výměna značně náročnější.

Fotocitlivý válec, obrazová jednotka (photoreceptor drum, drum unit, imaging unit)

Jedná se o válec, který je nejprve nabit pomocí koróny, následuje vykreslování bitmapy obrazu pomocí laseru. Následně je na něj z tonerového válce přenesena vrstva toneru, která je poté přitažena záporně nabitým válcem pod papírem na papír. Povrch tohoto válce je velmi jemný a jakékoliv poškození se okamžitě promítne v kvalitě obrazu.

Zapékací olej (fuser oil)

Jedná se o substanci, která se používá k promazávání zapékacího válce. Slouží k prevenci proti přichycení toneru na válec a poškozování tak dalších tisknutých stránek.

Čistič zapékacího válce (fuser cleaning pad)

Jedná se o součást, která stírá případné nečistoty a toner ze zapékacího válce. Tuto součástku je nutno periodicky vyměňovat a bývá taktéž potírána zapékacím olejem.

Ozónový filtr

Jelikož využívají tiskárny korónu k vytvoření náboje, vzniká množství ozónu, který je v nadměrném množství nebezpečný pro okolí a vytváří nepříjemný zápach. K odchycení plynů mívají tiskárny namontované mřížky s filtrem na výstupu větrání.

Zapékací jednotka (fuser unit)

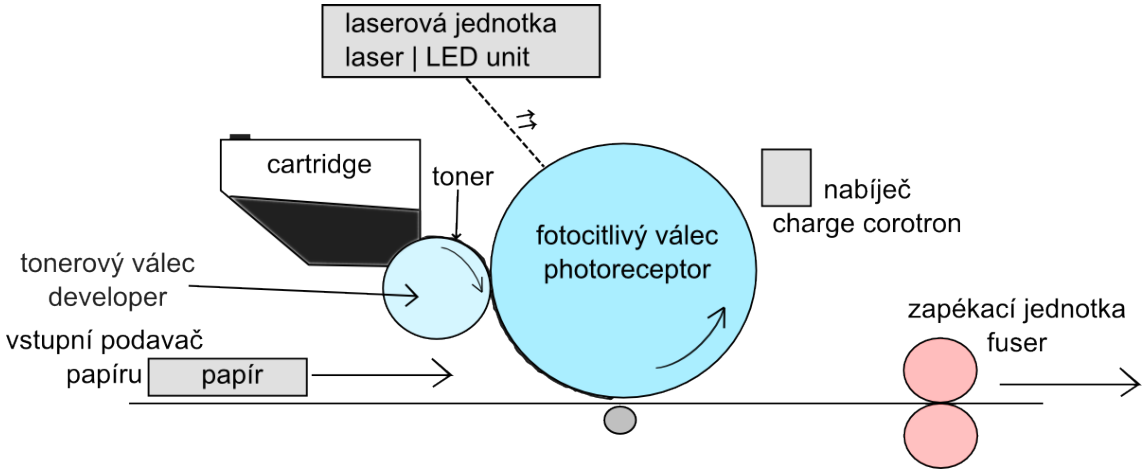
Sestává ze dvou válců, mezi nimiž putuje papír. Na papír s pokrývkou toneru je aplikována vysoká teplota, čímž dochází k zapečení toneru na papír. Zapékání toneru je poslední částí laserového tisku.

Odpadní nádobka (waste toner box)

Je nádobka, do které je sesypáván všechen přebytečný toner, který se čistí ze součástí, kde nemá co dělat. Jedná se například o válce zapékací jednotky, nebo hlavně fotocitlivé válce.

Přenosový pás (transfer belt, transfer roll, transfer unit)

Je součást, která zajišťuje pohyb papíru napříč všemi cykly potřebnými k tisknutí.



Obrázek 2.5: Schéma principu laserového tisku

2.2.3 Spotřební materiály inkoustových tiskáren

Jedná se o typy tiskáren, jejichž hlavní předností je nutnost menší a jednodušší údržby oproti tiskárnám laserovým. Obsahují pouze základní spotřební materiál, díky jednodušší metodě tisku.

Inkoust

Je tekuté médium používané k tisknutí. Inkoust je vždy přítomen v cartridge, která může být dvojího typu:

- s vestavěnou tiskovou hlavou, kdy se mění celá cartridge včetně tiskové hlavy. Pokud se tisková hlava poničí nebo přischne, je možno jednoduše vyměnit cartridge za jinou,
- bez vestavěné tiskové hlavy, kdy je cartridge pouze nádobkou na inkoust. Výhodou je možnost doplňování inkoustu, nevýhodou náročnější oprava v případě poškození tiskové hlavy.

Novým trendem je také využití tankového systému. Jedná se o nádobky umístěné většinou na boku tiskárny, uchovávající inkoust. Tyto nádobky lze jednoduše dolévat. Není tak potřeba vyměňovat či doplňovat cartridge, čímž neztratíme v případě poškození záruku a neušpiníme se.

2.2.4 Shrnutí

Každé zařízení je unikátní a obsahuje zcela rozdílné součásti, které je nutno po určitém čase vyměňovat. Naštěstí však zařízení obsahují čidla či počítadla, jenž umožňují určit opotřebovanost dané součásti. Měřením těchto stavů pak můžeme provádět jednotné a efektivní monitorování. V následující kapitole si popíšeme, jak tyto stavy získávat.

2.3 Způsoby získávání dat z tiskáren

Síťové tiskárny nám poskytují síťové služby, které můžeme využít k vzdálenému monitorování. Jako většina síťových zařízení i síťové tiskárny obsahují SNMP agenta, jenž nám

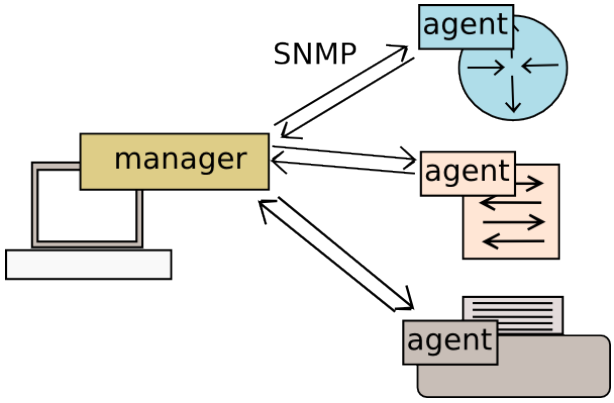
umožňuje monitorovat data pomocí protokolu SNMP. Pro obyčejné uživatele pak tiskárny obsahují webový server, pomocí něhož může být tiskárna konfigurována i zjišťován její stav. Jedná se tedy o protokol HTTP. Nyní si oba protokoly detailně popíšeme.

2.3.1 SNMP

Simple Network Management Protocol [3, 4] (protokol pro správu zařízení) představuje několik standardů pro monitorování a konfiguraci zařízení. Jedná se o síťový protokol, databázové schéma (MIB) a datové objekty. Tento protokol podporuje většina síťových zařízení, což nám dává možnost monitorovat jejich stav.

Síťový protokol

Jedná se o asynchronní protokol, založený na modelu klient-server, definován na aplikační vrstvě. Klient je zařízení, na kterém běží SNMP utilita (pro další zmiňování nazývaná manager), která zasílá dotazy serveru, na kterém běží software zvaný agent. Agent naslouchá na portu 161, manager na portu 162.



Obrázek 2.6: Architektura SNMP v síti

Základní typy zpráv:

- *get-request* - získání informace ze zařízení pomocí přesné adresy daného objektu (OID),
- *get-next-request* - získání informace z následujícího OID, které se nachází hned po zadaném OID. Umožňuje tak postupné procházení celým hierarchickým stromem,
- *set-request* - změna dané hodnoty objektu agenta (konfigurace).

Protokol byl vyvinut celkem ve třech verzích:

- verze 1 - autentizace probíhá pomocí *community stringu*, který slouží jako heslo. Toto heslo je jako obyčejný text přenášeno v každém dotazu a je tak snadné síť odposlouchávat a heslo získat,
- verze 2 - obsahuje navíc nový složitý zabezpečovací systém. Implementuje další typy zpráv, například v podobě dávkového dotazu, kdy jeden paket obsahuje dotaz na více OID, čímž je značně zmírněný přenos po síti,

- verze 2c - obsahuje všechny funkčnosti verze 2, ale používá zabezpečovací systém *community stringů* verze 1,
- verze 3 - tato verze je hojně využívána i pro nastavování, protože přibyly nové funkce pro autentizaci a šifrování, jimiž lze komunikaci zabezpečit.

MIB

Management Information Base je hierarchická stromová databáze, která popisuje objekty, jenž mohou být zjišťovatelné či nastavovatelné na zařízeních pomocí protokolu SNMP. Zařízení, které podporuje SNMP protokol, obsahuje různé podstromy, se kterými lze v rámci SNMP pracovat (získávat, nebo nastavovat jejich hodnoty). Objekty v těchto databázích mohou být jak koncové hodnoty, tak pouze označení dalšího podstromu, jenž obsahují další objekty. Tyto objekty jsou popsány pomocí notace *ASN.1*. Každý z těchto objektů je adresovatelný unikátním popisem OID (object identifier). Tento popis má formát čísel oddělených tečkou (1.3.6.1.2.1.43 - *Printer MIB*), kde každé číslo určuje položku v dané stromové úrovni. Podstromy objektů můžeme rozdělit podle proprietárnosti. Buďto jsou implementovány jedním výrobcem (enterprise větev 1.3.6.1.4), nebo jsou určeny pro všechny.

Printer MIB Tato databáze [5] byla vyvinuta za účelem jednotného získávání objektů tiskáren různých výrobců. Její OID je 1.3.6.1.2.1.43. Bohužel však výrobci *Printer MIB* podporují jen z části, a další nezbytné objekty specifikují ve své vlastní proprietární MIB, která se nachází v enterprise části (1.3.6.1.4), kde si organizace mohou zakládat své vlastní MIB. Problém poté nastává v parsování více MIB různých výrobců, kdy je potřeba pro každou MIB navrhnout jedinečný přístup k parsování a interpretaci informací. Tuto databázi hojně využívaly programy zkoumané výše. Pro základní získávání dat ji použijeme také.

System MIB Tato databáze je součástí *MIB II for TCP Based internet* [6] a byla vyvinuta za účelem získávání společných informací pro všechna síťová zařízení. Její OID je 1.3.6.1.2.1.1 a obsahuje objekty, jako například: název zařízení, umístění, doba spuštění, ... Tuto databázi taktéž využívaly programy zkoumané výše. Pro získávání obecných systémových informací ji využijeme také.

Brother MIB Tato databáze je příkladem proprietární databáze firmy Brother. Tiskárny jiných výrobců tedy tuto MIB nepodporují. Její OID je 1.3.6.1.4.1.2435 a obsahuje veškeré systémové i spotřební informace, jenž může zařízení nabídnout.

Získávání údajů

Pokud chceme získat informace ze zařízení, potřebujeme znát přesné číslo objektu OID. Tato čísla je možné přeložit do lidsky čitelné textové podoby tak, že získáme MIB databázi, která zastřešuje daný objekt (např. 1.3.6.1.2.1.43.10.2.1.4 - *prtMarkerLifeCount* - *Printer MIB*). Pokud chceme vypsat veškeré objekty jenž zařízení implementuje, musíme začít od kořene stromu, tedy od OID = 1. Pak pomocí SNMP příkazu *getNext* ze zařízení postupně vyčítáme veškeré objekty, které obsahuje.

Snmpget Je utilita, která nám umožňuje pomocí SNMP *get-request* dotazů získat hodnotu daného OID. Je potřeba specifikovat verzi protokolu, *community string*, OID a adresu

zařízení, na kterém naslouchá SNMP agent. Tuto utilitu lze používat k získání konkrétních údajů ze všech zařízení.

Snmpgetnext Je utilita stejná jako předchozí snmpget s tím rozdílem, že místo *get-request* dotazů používá *get-next-request* dotazy. Nezískává tedy daný OID, ale hodnotu jeho nástupce ve stromu. Pomocí ní lze proiterovat celou databází, jenž daný agent podporuje.

Snmpwalk Tato utilita se chová obdobně jako předchozí s tím rozdílem, že získá všechny hodnoty podstromu daných kořenovým OID. Pokud bychom například chtěli získat údaje *Printer MIB* databáze, uvedli bychom jako kořenové OID - OID *Printer MIB* databáze.

```
SNMPv2-SMI::mib-2.43.11.1.1.9.1.1 = INTEGER: 39
SNMPv2-SMI::mib-2.43.11.1.1.9.1.2 = INTEGER: 81
SNMPv2-SMI::mib-2.43.11.1.1.9.1.3 = INTEGER: 72
SNMPv2-SMI::mib-2.43.11.1.1.9.1.4 = INTEGER: 81
SNMPv2-SMI::mib-2.43.12.1.1.4.1.1 = STRING: "black"
SNMPv2-SMI::mib-2.43.12.1.1.4.1.2 = STRING: "cyan"
SNMPv2-SMI::mib-2.43.12.1.1.4.1.3 = STRING: "magenta"
SNMPv2-SMI::mib-2.43.12.1.1.4.1.4 = STRING: "yellow"
```

Obrázek 2.7: Ukázka výpisu z programu Snmpwalk

Probouzení tiskáren

Při testování jsem narazil na problém, kdy se některé tiskárny nacházejí v úsporném režimu a je nutné je před samotným dotazováním probudit (týká se pouze OKI). Pokud by se tak nestalo, tvářily by se, jako by SNMP vůbec nepodporovaly. Vyzkoušel jsem dvě metody. První pomocí ICMP paketů a druhou pomocí navázání TCP spojení na web. server tiskárny. S tímto probouzením je nutno při vytváření modulů počítat.

Výzkum

Každou ze zkoumaných tiskáren (viz tabulka testovaných tiskáren 2.1) jsem podrobil výzkumu MIB databází, jenž podporují. Zjistil jsem, že získávané hodnoty mohou být dvou typů.

- Statické - hodnoty, které mají staticky stanovený OID. Přesně víme, jaký typ hodnoty dostaneme a co je to za hodnotu. Pro získávání hodnoty používáme typ zprávy *get-request*. Pokud tiskárna hodnotu nepodporuje, jednoduše nám nevrátí její hodnotu.
- Stromové - hodnoty, které jsou získávány z podstromu. Nikdy nevíme, kolik položek v podstromu bude a pod jakým budou indexem. Pokud ale index získáme, víme, jaké položky tento podstrom obsahuje. Můžeme se na ně odkázat přidáním dalších indexů těchto položek. Víme, jaké hodnoty budou obsahovat a co je to za položky. Pro získání ale potřebujeme nejdřív získat seznam indexů, jenž získáme pomocí iterování přes *get-next-request*. Příkladem jsou seznamy aktuálních varování tiskáren (z *printer MIB*). Základní OID je 1.3.6.1.2.1.43.18.1.1. Z něj získáme seznam podstromů (například 1.3.6.1.2.1.43.18.1.1.2 a 1.3.6.1.2.1.43.18.1.1.90). Nyní již víme, že připojením .2 získáme úroveň varování, připojením .8 popis, ...

Nyní si popíšeme jednotlivé databáze MIB, které jsem na tiskárnách zkoumal.

System MIB Databáze byla podporována všemi zkoumanými tiskárnami a můžeme ji tedy využít pro získávání obecných dat týkajících se názvu, umístění, popisu a době spuštění. Tyto položky jsou získávány statickou metodou, tedy předem známe i jejich název.

Printer MIB Databáze byla podporována všemi tiskárnami. U tiskárny Canon však neposkytovala žádné podstatné informace. Z této databáze vyčítáme především varování a hodnoty spotřebních materiálů. Tyto položky lze získávat pouze pomocí metody stromové. Při získávání hodnot spotřebních materiálů jsem narazil na problém, že ne všechny položky obsahují smysluplnou hodnotu, která by nám umožnila výpočet procenta. Některé obsahovaly záporné hodnoty, což značí, že hodnotu nelze určit, nebo že je dostupná (jen nevíme jakou má hodnotu). Výpočet na procenta provádím pomocí indexů 8 (maximální kapacita) a 9 (aktuální hodnota). Název spotřebního materiálu je dán indexem 6 (popis). U získávání varování je postup obdobný.

Printer Brother Databáze podporována pouze tiskárnami firmy Brother. Tyto tiskárny nepodporovaly výpisy všech spotřebních materiálů pomocí *Printer MIB*. Byla použita díky vedoucímu bakalářské práce, který reverzním inženýrstvím rozparsoval jednotlivé hodnoty, které jsem v práci mohl dále využít. Umožňuje získat hodnotu údržby 1.3.6.1.4.1.2435.2.3.9.4.2.1.5.5.8, jenž obsahuje hexadecimální string. Tento string obsahuje řetězec sedmic Bytů. Z něj je nutno vyparsovat hodnoty spotřebních materiálů a ručně dodat, o jakou se jedná. První 3 Byty obsahují index, zbývající hodnotu. Nutno dodat, že každá tiskárna obsahuje jiné sedmice. Indexy se však v mnoha z nich překrývají. Na základě toho lze přibližně odhadnout, co je spotřební materiál a co jen počítadlo. Například 0x69,70,71,72 jsou indexy pro spotřební materiály tonerů (CMYK) (pro ověření významu indexů lze najít a porovnat hodnoty na webových stránkách tiskáren a na základě rovnosti jednoznačně určit jejich typy).

2.3.2 HTTP

Hyper Text Transport Protocol [7] je aplikační, spojovaný, bezstavový internetový protokol založen na komunikaci klient-server. Server je zařízení, které naslouchá na portu 80 nezabezpečené, případně 443 zabezpečené komunikace. Na serveru je spuštěna služba webového serveru - například Apache, IIS, Nginx, nebo GWS. Klientem je zařízení, jenž umožňuje komunikaci pomocí protokolu HTTP, která se nejčastěji používá pro přenos webových stránek (i jiných souborů pomocí rozšíření MIME). Jelikož je protokol bezstavový, používá se k identifikaci klienta a jeho relace *Session ID*, které klient musí odesílat spolu s požadavkem, aby byl identifikován.

Komunikace

Klient adresuje server a soubor pomocí URI adresy na aplikační úrovni. Komunikace probíhá pomocí metody dotazování. Klient serveru odešle zprávu a server zareaguje. Nejznámějšími typy zpráv, určené v hlavičce protokolu jsou:

- GET - požadavek na stažení uvedeného objektu,
- HEAD - požadavek pouze na hlavičku daného objektu (nestahuje data, jen metainformace),

- POST - nahrazení požadavku GET v případě, že nechceme, aby data dotazu byla viditelná v URL.

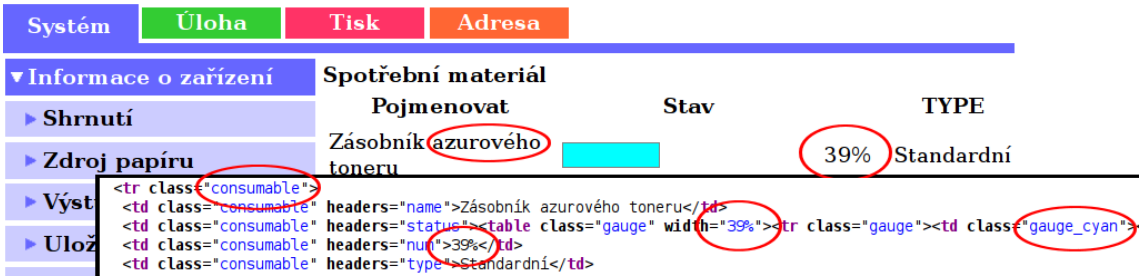
Odpověď od serveru v hlavičce nese stavový kód a případně další informace s daty v těle zprávy. Tyto stavové kódy informují například o přesměrování, chybě nalezení obsahu, nebo pokud vše proběhlo v pořádku a jsou dostupná data.

HTML

HyperText Markup Language je značkovací jazyk, který je používán pro serializaci dat a tím přenos webových stránek. Na straně klienta běží v prohlížeči interpret, který danou stránku (spolu s CSS, ...) (přijatou protokolem HTTP/s) zobrazí s grafickým rozhraním. Soubor HTML obsahuje mnoho informací, z nichž jsou nejpodstatnější informace o entitě - *class* a *id*. Ty nám určují jednotlivé uzly, jenž identifikují data (hodnoty spotřebních materiálů, ...).

Využití

Většina tiskáren obsahují webový server pro přehlednou konfiguraci a zobrazování stavu uživatelům. Z toho důvodu je protokol HTTP druhá varianta, která půjde využít pro získávání dat. Problémem bude na rozdíl od SNMP parsování dat z HTML stránek. Každá HTML stránka je totiž jiná a většinou musí dojít k autentizaci před samotným prohlížením dat. Příklad webové stránky se zobrazeným HTML souborem je níže (2.8).



Obrázek 2.8: Ukázka web. stránky a jejího HTML kódu pro parsování informací

2.3.3 Shrnutí

Nyní máme dvě metody použitelné pro získávání dat z tiskáren. Oproti konkurenčním aplikacím máme další metodu jakým způsobem získávat data, která nebudou dostupná pomocí SNMP. Nevýhoda HTTP metody (nebyla navržena k těmto monitorovacím účelům) je však dána složitějším parsováním a přenosem dalších zbytečných dat po síti, čímž vzroste datový tok. Vzhledem k nízké frekvenci dotazování a přenášení pouze textového obsahu je však toto číslo téměř zanedbatelné. Nyní se nabízí otázka jak jednotně získávat informace pomocí obou metod, aby byl návrh aplikace čistý a jednoduše použitelný pro uživatele.

Kapitola 3

Návrh řešení

Po provedení výzkumu můžeme přistoupit k návrhu aplikace. Nejprve si specifikujeme požadavky a poté aplikaci navrhneme.

3.1 Specifikace požadavků

Abychom mohli navrhnout výsledné řešení, musíme si popsat požadavky, které jsou kladeny na výslednou aplikaci. Tyto požadavky můžeme rozdělit do dvou kategorií. Požadavky základní, jenž jsou dány zadáním a požadavky odvozené, které vyplývají z předchozího výzkumu a umožňují aplikaci vytvořit lépe.

3.1.1 Základní požadavky

Základní požadavky jsou požadavky vyplývající ze zadání bakalářské práce. Jedná se o ty požadavky, které jsou nezbytně nutné k vytvoření hrubé kostry výsledné aplikace.

Monitorování tiskáren

Tento požadavek zahrnuje získávání dat z tiskáren různých výrobců. Jak dokázal předchozí výzkum, stále neexistuje aplikace, která by dokázala sledovat spotřební materiály tiskáren všech typů různých výrobců. Na základě toho jsem tedy navrhl metody získávání pomocí protokolů SNMP a HTTP, jenž by teoreticky měly pokrýt veškeré moderní typy tiskáren.

Ukládání dat

Data získaná monitorováním je potřeba ukládat. U této aplikace nám nezáleží na rychlosti vkládání. Vždy se bude jednat o jednoduchá data, jejichž počet vkládání bude závislý na četnosti sledování proměnných z tiskáren a jejich počtu. Při předpokladu 500 tiskáren v podnikové síti, kdy z každé sledujeme 20 spotřebních materiálů jednou za hodinu, činí počet vkládání přibližně 3 za sekundu. Mnohem důležitější je poté doba dotazování, kdy pro získání celého přehledu musíme ve zlomku sekund získat všech 10000 informací. Z hotových řešení se dají použít třeba relační, nebo nerelační databáze. Sáhnout se dá i po řešení vlastním v podobě ukládání do textových souborů.

Zobrazování dat

Uložená data je potřeba zobrazit správcům sítě pomocí webového nástroje tak, aby byla přehledná a umožnila jim provádět efektivní monitorování a kontrolu tiskáren. Webové rozhraní musí být přehledné, stručné a výstižné, jelikož počet tiskáren, jenž může být spravován, může být velmi vysoký.

3.1.2 Požadavky odvozené

Odvozené požadavky jsou požadavky, které jsem si stanovil při provádění výzkumu. Díky průzkumu konkurenčních aplikací můžeme využít jejich silných stránek, které do naší aplikace zakomponujeme taktéž a naopak vyvarovat se, nebo zlepšit jejich stránky slabé.

Řazení dat a skupinování

Z aplikací můžeme odvodit metody, které umožňují výstup aplikace přizpůsobit tak, aby byl přehledný a vypovídající. Jedná se o metody řazení, jenž nám umožní například zobrazit tiskárny sestupně podle počtu varování, které obsahují, nebo skupinování, jenž umožňuje zařazení tiskáren do určitých skupin (podle společných vlastností) a zobrazení pouze tiskáren, které patří do dané skupiny (například dle umístění v patře). Obě metody pak vedou k zefektivnění činnosti správce.

Zobrazení volitelných dat

Žádná z aplikací neumožňovala k tiskárnám a jejich spotřebním materiálům přidat vlastní popisky, jenž by dokázaly lépe vystihnout účel dané vlastnosti. Díky tomu by si správce sítě mohl k spotřebním materiálům dopsat například katalogové číslo, aby mohl spotřební materiál ihned objednat.

Ukládání časové řady

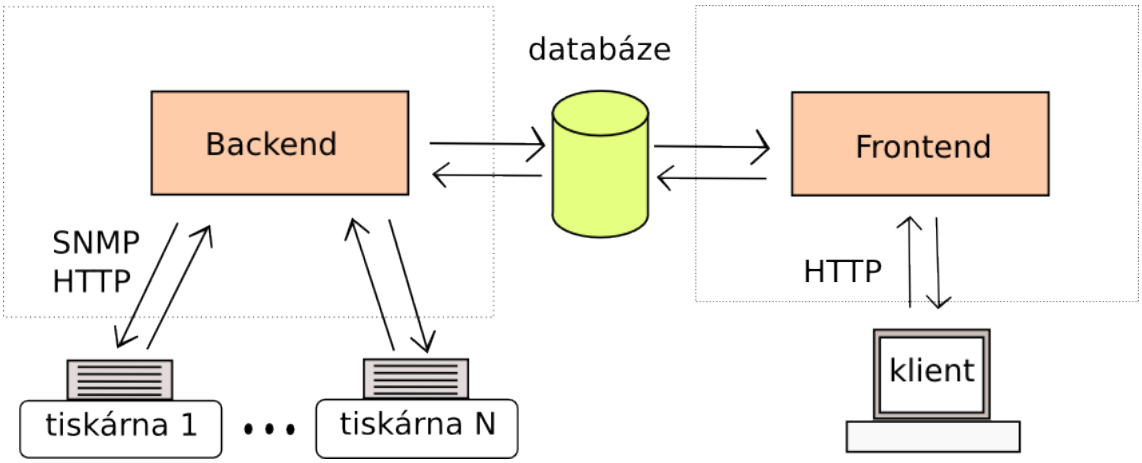
Žádná z aplikací taktéž neumožňovala ukládání dřívějších dat pro jejich zpětné sledování. Ukládání bylo jednorázové a nebylo možné uložit více hodnot stejného typu naměřených v různé časy. Při přidání této vlastnosti by správcům bylo umožněno prohlížet si stavy spotřebních materiálů zpětně v čase a vykreslovat je do přehledných grafů, nebo například vypočítat rychlosti úbytků těchto spotřebních materiálů a predikovat výměny do budoucna.

Architektura

Všechny sledované aplikace fungovaly samostatně na jednom stroji. Nebylo možné aplikaci odděleně spustit tak, aby jedna sbírala data a druhá je zobrazovala. V praxi to tedy znamená - mít na jednom stroji zprovozněnou databázi, i prostředí pro frontend a backend. Tato architektura je ovšem nedostatečná, pokud máme v síti servery, jenž jsou jednoúčelově zaměřeny na tyto úkony. Cílem tedy bude navrhnout aplikaci tak, aby všechny její části mohly běžet na fyzicky jiných serverech a zároveň zachovat možnost běhu souběžně na právě jednom.

3.2 Základní návrh aplikace

Na základním schématu můžeme vidět rozdělení aplikace na 2 části. První z nich - backend, který sbírá a ukládá data a druhou - frontend, jenž data načítá a zobrazuje. Obě části běží odděleně a lze je tak spustit na oddělených serverech. Mají společnou pouze jednu věc. Databázi a hodnoty s nimiž musí pracovat.



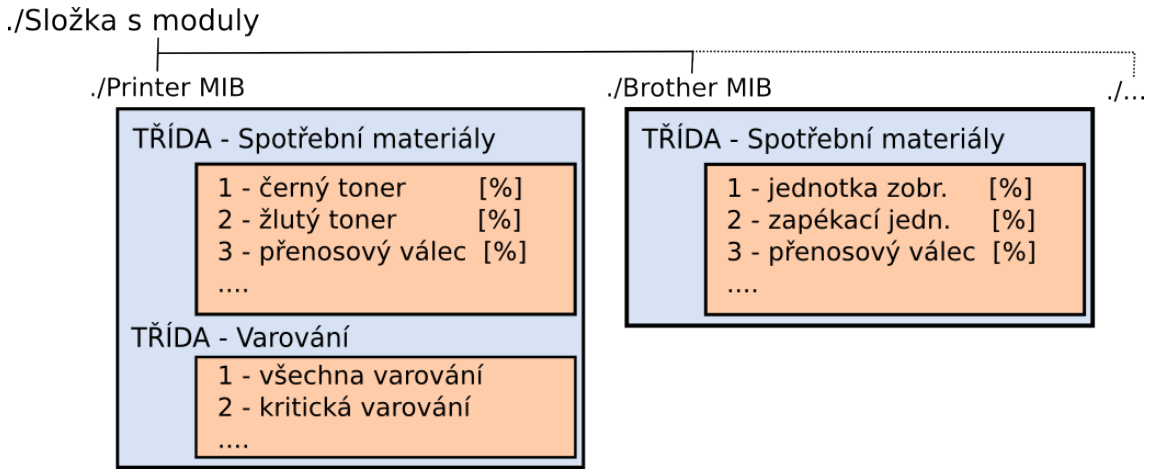
Obrázek 3.1: Zobecněný návrh programu

Nyní popíši nejhlavnější části aplikace.

3.2.1 Získávání dat

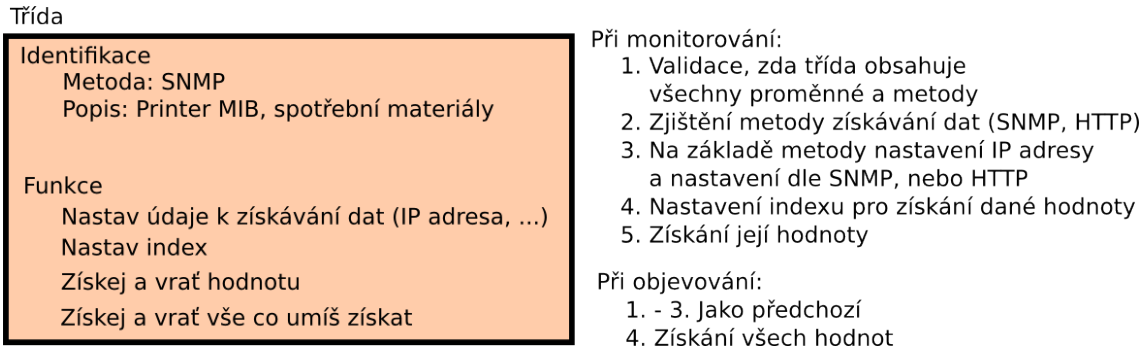
Z předchozího výzkumu víme, že pro získávání dat jsou použity protokoly SNMP a HTTP. Jakým způsobem navrhnout získávání hodnot tak, aby bylo víceúčelové (použitelné pro více druhů tiskáren, nikoliv pouze pro jeden typ) a přidávatelné (uživatel by si mohl navrhnout vlastní metody, pokud by mu současně implementované přišly nedostatečné)?
Navrhl jsem řešení pomocí modulů a tříd. Tyto moduly (soubory v daném programovacím jazyce) obsahují třídy, jenž mají definované jednotné rozhraní a samotnou implementaci aplikace nevidí (blackbox). Moduly jsou tedy přidávatelné i vyměňovatelné mezi uživateli. Aplikace zjišťuje, jakou metodou třídy data získávají, zda HTTP, nebo SNMP a podle toho jim při inicializaci předává přihlašovací údaje pro autentizaci (SNMP - community string, verze, HTTP - přihl. jméno, heslo).
Moduly si tedy uživatelé mohou vyměňovat a dopisovat vlastní, při zachování rozhraní jejich tříd. Jakým způsobem však získat konkrétní proměnnou z tiskárny? Řešení je v podobě indexů, kdy třída obsahuje pole indexů v intervalu `<0, N>`, na které se aplikace může vyptat. Každý index pak představuje jednu hodnotu (př: index 0 = černý toner, index 1 = žlutý toner, ...).
Uživatel ovšem potřebuje jednoduchou metodu jak zjistit, které proměnné lze pomocí dané třídy získat. K tomu je součástí rozhraní každé třídy funkce, která vrátí veškeré údaje, jenž z tiskárny může získat. Na základě nich se uživatel rozhodne, které funkce chce monitorovat a přidá je do konfiguračního souboru (součástí aplikace utilita, jenž automaticky prochází všechny moduly a vytváří konfigurační soubor s tiskárnami, viz 4.2.2). Pojďme se tedy podívat na názorné příklady.
Na následujícím obrázku 3.2 lze vidět, vztah mezi moduly, třídami a indexy. Moduly jsou

soubory *Printer MIB* a *Brother MIB*. Třídy pak umožňují získat proměnné, jenž spolu logickým způsobem souvisí (zde podstromy daných MIB databází). Nastavením indexu lze poté získat danou hodnotu (modul *Printer MIB*, třída Varování, index 1 - všechna varování).



Obrázek 3.2: Struktura modulů

Všechny třídy v modulech musí splňovat jednotné rozhraní. Toto jednotné rozhraní aplikace využívá pro získávání dat. Vizte na dalším obrázku 3.3 níže.



Obrázek 3.3: Rozhraní modulů

Proměnná metoda ve třídě určuje, jakým způsobem třída získává data a jak ji nakonfigurovat při instancianci. Popis třídy popisuje způsob získávání dat či výčet získávaných hodnot. Metody tříd slouží k nastavení autentizačních údajů, získání všech hodnot, nastavení indexu a získání hodnoty na nastaveném indexu.
Aplikace tedy může načíst N modulů s N různými třídami. Každá z těchto tříd se specializuje na určitá data. Všechny z nich však obsahují stejné rozhraní, aby pomocní něj mohly být z tiskáren data získávány.

3.2.2 Konfigurace

Program je potřeba dále nakonfigurovat. Jako formát jsem vybral jazyk XML, jenž je pro tyto účely běžně používán. Vybral jsem ho hlavně pro jeho přehlednost a jednoduchost. Konfiguraci můžeme rozdělit do dvou sekcí.

Sekce `<general>`

Obsahuje hlavní konfiguraci potřebnou pro chod programu. Jedná se o cestu k modulům tiskáren (viz 3.2.1), přihlašovacích údajů k databázi (viz 3.2.3), výchozí nastavení pro SNMP (*community string*, verze) a výchozí nastavení pro HTTP (přihlašovací jméno, heslo). Z důvodu bezpečnosti nejsou tato data předávána v argumentech programu, aby jiný uživatel nemohl snadno získat přihlašovací údaje běžným vypsáním procesů běžících v systému.

Sekce `<printers>`

Obsahuje konfigurace tiskáren (viz podkapitola 4.2.2), jenž chceme sledovat. Pokud chce uživatel sledovat tiskárnu, musí zde přidat její konfiguraci. Tu také dělíme na dvě části.

Popis Jedná se především o její IP adresu (`<IP>`), nebo DNS jméno (`<DNS>`) a její unikátní jméno pro identifikaci v databázi (`<name>`). Atributem `<active>` říkáme, zda chceme tiskárnu sledovat (získávat z ní data), nebo si pouze prohlížet její již získaná data ve frontendové části. Dalšími atributy jsou položky potřebné pouze pro její podrobný popis. Jedná se o jméno výrobce (`<vendor>`), typ (`<type>`), umístění (`<location>`), popis (`<note>`) a skupinu (`<group>`, pro zařazení tiskárny do skupiny a zpřehlednění zobrazení dat). Atribut `<time>` určuje, jaký časový interval je použit v opakovaném získávání všech dat, pokud nebude uvedeno explicitně u sledované hodnoty (viz 3.2.4). Dále lze určit výchozí nastavení SNMP a HTTP pro všechny její sledované hodnoty, pomocí tagů `<snmpConf>` a `<httpConf>`.

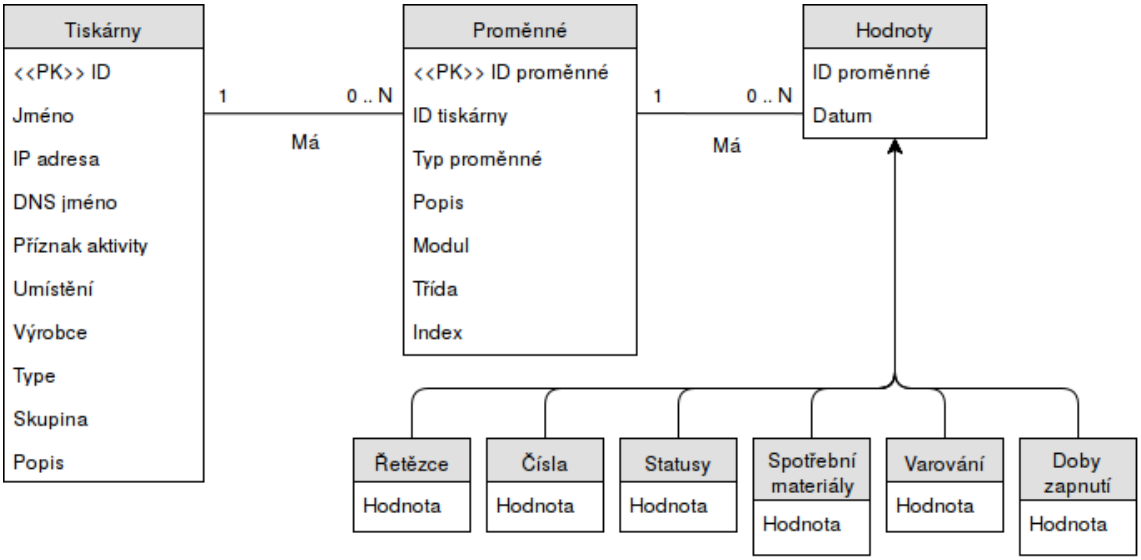
Sledované hodnoty Jedná se o výčet Nx `<snmp>` a Nx `<http>` tagů, jenž v sobě obsahují údaje pro získávání dané hodnoty z tiskárny. Název tagu říká, která metoda bude použita - zda SNMP, nebo HTTP. Uvnitř něj se nachází název modulu a jeho třídy (obě v tagu `<source>`, syntaxí *modul.třída*). Jedná se například o modul *printerMIB* a třídu *Consumables* (spotřební materiály). Danou hodnotu získáme pomocí tagu index (`<idx>`), jenž jednoznačně určuje danou hodnotu v pořadí. Hodnotu lze získávat opakovaně pomocí nastavení tagu `<time>`, ve kterém specifikujeme časový interval mezi jednotlivými získáváními (viz 3.2.4). Dále lze zadat název proměnné `<name>` a její obšírný popis `<note>`, jenž (pokud jsou zadány) přepíší názvy získané pomocí dané třídy.

3.2.3 Databáze

Jakmile získáme data, potřebujeme je uložit. K uložení máme jak statická data (obecné údaje), tak hodnoty získávané v čase, které musí být rychle vyhledatelné (frontend).
K dispozici máme 3 možnosti ukládání:

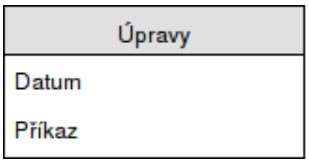
- vlastní řešení (.txt) - složitá implementace, nutnost načítat a upravovat celé textové soubory, jenž by časem znamenalo značné zpomalení,
- nerelační databáze - umožňuje vkládání různých dat, nezávisle na schématu, ovšem za určitou cenu, nezajišťuje stálou konzistenci dat,
- relační databáze - statické schéma databáze, neustále zajištěna konzistence.

Jelikož mají námi sledovaná data statické schéma, vybral jsem databázi relační, se kterou mám také největší zkušenosti.
Po výběru relační databáze jsem navrhl konceptuální model dat.



Obrázek 3.4: Konceptuální návrh databáze

Tento model je navržen tak, aby nebylo potřeba při vkládání či načítání používat jakékoliv JOIN operace, jenž by dotazy zpomalily. Údaje tabulek *Tiskárny* a *Proměnné* jsou statické a jsou ukládány pouze na začátku spuštění programu. Každá proměnná je jiného typu (určeno pomocí sloupce *Typ proměnné*), pomocí něhož lze určit, v jaké tabulce se hodnoty nachází. K tomu si stačí zapamatovat *ID proměnné*, aby šlo provádět jakékoliv vkládací, či doptávací dotazy bez nutnosti JOIN operací.
Aby šlo lehce dohledávat změněné údaje, slouží jako případná záloha další tabulka jenž obsahuje veškeré dotazy, které modifikovaly statická data (tento nápad pochází od vedoucího bakalářské práce).



Obrázek 3.5: Tabulka pro zálohování statických dat

3.2.4 Opakované získávání dat

Jakmile víme, jak budeme získávat data, musíme zajistit, aby byla data získávána opakovaně. Tento úkol provádí kalendář událostí, jenž obsahuje seřazené pole událostí podle času jejich spuštění. Pokud aktuální čas je roven času nejdřívější události v poli, událost je spuštěna a čeká se, dokud čas spuštění další události není roven času aktuálnímu. Zároveň je nastaven nový čas spuštění prováděné události a je znovu zařazena podle tohoto času zpět do pole. Abychom mohli sledovat méně důležité události méně často, nebo naopak, lze každé události nastavit různý čas opakování, jenž se nastavuje v konfiguračním souboru pomocí tagu `<time>`. Tuto možnost konkurenční aplikace taktéž nenabízely.

3.2.5 Zobrazení dat ve frontendu

Frontendová část musí být dle zadání implementována jako webové rozhraní. Toto rozhraní musí být výstižné a jednoduché. Správce by z něj měl vyčíst ty nejpodstatnější informace nutné k zajištění provozuschopného stavu tiskáren. Mělo by umět řazení dat, stejně jako jejich shlukování pomocí argumentu skupin. Ukázka výsledné implementace webového rozhraní se nachází v příloze C.

Stránka skupin (hlavní strana)

Zobrazuje tabulku se skupinami, do kterých tiskárny patří, a jejich metainformace. Tyto metainformace říkají, kolik aktivních a neaktivních tiskáren skupina obsahuje, kolik varování obsahují všechny tiskárny náležící dané skupině a kolik spotřebních materiálů je v intervalech `<0,20)%`, `<20,40)%`, `<40,60)%`, `<60,80)%`, `<80,100)%`. Tento přístup umožní správci okamžitě zjistit, zda se ve skupině nachází tiskárny s varováními, nebo spotřebními materiály, které zrovna dochází. Veškeré sloupce lze řadit.

Stránka skupiny

Po rozkliknutí dané skupiny z hlavní stránky se zobrazí tabulka veškerých tiskáren náležících dané skupině. Tyto tiskárny jsou také v řaditelném seznamu obsahující DNS jméno, výrobce, typ, počet varování a intervaly hodnot spotřebních materiálů, jako na hlavní stránce. Lze tedy okamžitě najít tiskárny s nízkými hodnotami spotřebních materiálů, nebo varováními.

Stránka tiskárny

Po rozkliknutí tiskárny ze stránky skupin lze vidět veškeré informace, jenž tiskárna poskytuje. Lze si prohlížet varování, spotřební materiály i veškeré statické údaje (IP adresa, umístění, ...).

Kapitola 4

Implementace

V této kapitole je popsán výběr implementačního jazyka, použitých knihoven a implementace backendové i frontendové části.

4.1 Výběr programovacího jazyka

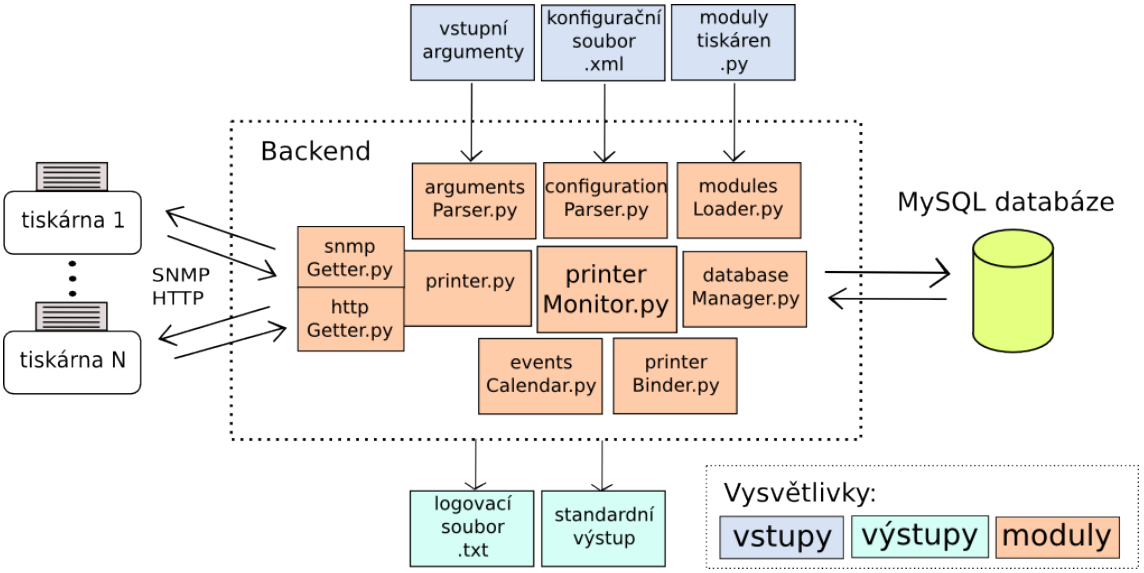
Při výběru programového vybavení bylo přání vedoucího mé bakalářské práce, aby byl zvolen stejný jazyk jak pro backend, tak frontend. Nabízelo se několik základních možností:

- Python,
- PHP,
- C#.

Všechny tři programovací (C#) a skriptovací (Python, PHP) jazyky nabízejí možnosti jak pro napsání backendu, tak frontendu.
C# jsem zamítl z toho důvodu, že je to kompilovaný jazyk. Přidávání modulů, jenž by mohl uživatel jednoduše přidávat, aniž by při tom kompiloval kód, by nebylo jednoduše možné. Bylo by potřeba navrhnout vlastní jazyk a interpret. Taktéž nám nezáleží na rychlosti aplikace.
Zbývající PHP a Python jsou skriptovací jazyky a jsou na tom z pohledu výkonu téměř stejně. Vždy záleží na verzi a čistotě zdrojového kódu. V různých případech jeden předhání druhý [8]. Rozhodl jsem se tedy použít jazyk, se kterým mám větší zkušenosti, tedy Python, ve verzi 2.7.

4.2 Implementace backendu

Backendová část je sestavena z modulů. Každý modul je řešen jako třída, jenž se stará o jemu přidělenou část. Každý modul zároveň obsahuje svou vlastní třídu pro výjimky a instanci logovacího výstupu. Jádro (*printerMonitor.py*) je hlavní smyčka programu, ve které jsou tyto moduly instanciovány a volány jejich funkce. Struktura backendu je vidět na obrázku 4.1.



Obrázek 4.1: Podrobný návrh backendu

Dle tohoto návrhu jsou vykonávány kódy modulů (tříd) z hlavní smyčky programu (*printerMonitor.py*) v následujícím pořadí:

1. načítání a parsování všech vstupních argumentů (*argumentsParser.py*),
2. načítání a parsování konfiguračního souboru (*configurationParser.py*),
3. načítání všech modulů tiskáren (*modulesLoader.py*), jenž umožňují z tiskáren získávat data pomocí funkcí, jenž implementují. Tyto moduly obsahují třídy, jenž využívají pomocné moduly (*snmpGetter.py* a *httpGetter.py*),
4. vytvoření databázového spojení, validování databázového schématu, popř. vytvoření nového (*databaseManager.py*),
5. vytvoření kalendáře událostí, v němž se nachází opakované události, jejichž voláním jsou z tiskáren získávána data (*eventsCalendar.py*),
6. propojení konfigurace tiskáren s jejich moduly a databází a vložení sledované proměnné do kalendáře událostí (*printerBinder.py*),
7. spuštění kalendáře událostí (*eventsCalendar.py*).

Nyní si popíšeme význam jednotlivých modulů a jejich funkce.

4.2.1 Parser argumentů - `ArgumentsParser`

Parser argumentů načítá veškeré argumenty předané programu při jeho spuštění. Mezi hlavní předané argumenty patří:

- -c [] -> cesta ke konfiguračnímu XML souboru s dalším nastavením,
- -l [] -> cesta k výstupnímu logovacímu souboru (při neexistenci je vytvořen nový),

- –deletePrinters -> odstranění tiskáren z databáze, jenž již nejsou uvedeny v konfiguračním souboru (byly uloženy do databáze z předchozího spuštění programu, když se nacházely v konfiguračním souboru),
- –deleteArguments -> odstranění sledovaných hodnot z databáze, jenž již nejsou uvedeny v konfiguračním souboru u daných tiskáren,
- -v -> zapnutí debugovacího módu (ladící výpisy).

4.2.2 Parser konfigurace - `ConfigurationParser`

Parser konfigurace načítá konfigurační soubor[1], jehož struktura je popsána v podkapitole 3.2.2. Zvlášť jsou načítána obecná nastavení aplikace (uložení do asociativního pole) a nastavení tiskáren (uložení do pole).

Získávání konfigurace - `ConfigurationGetter`

Aby uživatel nemusel monitorování tiskáren konfigurovat ručně, vytvořil jsem pomocný skript *configurationGetter.py*, nacházející se ve složce *PriMoVi/printerMonitor/utils* (viz příloha B). Tento skript očekává následující argumenty:

- -a [] -> IP adresa tiskárny (není-li zadáno DNS jméno),
- -d [] -> DNS jméno tiskárny (není-li zadána IP adresa),
- -n [] -> unikátní jméno tiskárny pro identifikaci v databázi (např: Pri42),
- -m [] -> cesta k modulům, jenž z tiskáren získávají data,
- -c [] -> community string použit při získávání dat pomocí SNMP,
- -v [] -> verze SNMP použita při získávání dat pomocí SNMP. Při nezadání je zjištěna automaticky,
- -o [] -> výstupní soubor, kam se má konfigurace zapsat (je přidána na konec souboru)
- -M -> Vypíše i rozšiřující tagy, bez zjištěných hodnot. Tyto tagy musí doplnit uživatel, nebo je smazat.

Výhodou tohoto skriptu je, že uživatel nemusí o tom, jak aplikace zjišťování provádí vůbec nic vědět. Aplikace potřebuje pouze IP adresu tiskárny a její unikátní jméno. Zjištění verze SNMP či módu, jakým je nutno tiskárnu probouzet už skript obstarává sám. Na základě toho vybírá jen ty spotřební materiály, které těmito způsoby lze získat.
Skript je interaktivní a uživatele se ptá, zda chce daný spotřební materiál přidat do konfigurace. Výsledkem je poté korektní jím stanovená konfigurace, kterou lze jednoduše vložit do konfiguračního souboru mezi značky `<printers>`.

[1] Ukázky konfiguračních souborů lze najít v *PriMoVi/printerMonitor/config*, viz příloha B.

4.2.3 Načítač modulů tiskáren - `ModulesLoader`

Načítač modulů tiskáren na vstupu očekává cestu ke složce, která je předána v konfiguračním souboru. Tato složka obsahuje moduly v jazyku python (viz 3.2.1), používané k získávání dat z tiskáren. Tyto moduly mají společný předpis, jenž jsou tímto modulem validovány a uloženy do pole modulů a tříd. Defaultní cesta ke složce, jenž moduly obsahuje, je */src/printersModules*.

4.2.4 Databázový manager - `DatabaseManager`

Databázový manager logicky odděluje program od implementace databáze. Je zodpovědný za validování databázového schématu a v případě neexistence vytvoření nového. Uživatel se tak nemusí starat o správu databáze. Musí pouze vytvořit jednu prázdnou a údaje k ní předat do konfiguračního souboru. Tento modul dále vytváří a udržuje spojení s databází a provádí veškeré dotazy nutné pro vkládání, mazání, dotazování, nebo získávání dat.

4.2.5 Kalendář událostí - `EventsCalendar`

Kalendář událostí je modul, který periodicky spouští akce, jenž vedou k získávání dat z tiskáren. Je spuštěn po veškerých inicializacích a běží dokud uživatel program nezastaví. Veškerá jeho činnost je popsána v podkapitole 3.2.4.

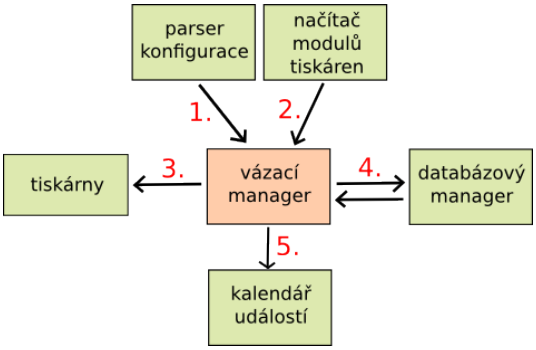
4.2.6 Tiskárny - `Printer`

Tato třída reprezentuje dané tiskárny v řeči počítače. Každá načtená tiskárna z konfigurace je uložena do jedné instance této třídy. Obsahuje veškeré své statické údaje (IP adresa, název, výrobce, ...) i seznam metod potřebných k získávání dat. Odkazy na tyto metody jsou poté předány kalendáři událostí.

4.2.7 Vázací manager - `PrinterBinder`

Vázací manager je zodpovědný za provázání všech předchozích modulů. Zajišťuje svázání konfigurace s moduly tiskáren, jejich synchronizaci s databází a vložení událostí do kalendáře událostí. Cyklicky pro každou tiskárnu provádí:

1. uložení statických údajů tiskárny v konfiguraci do třídy Printer (IP adresa, název, ...),
2. dle konfigurace u každé získávané hodnoty dané tiskárny - načtení třídy a indexu (z `ModulesLoader`), pomocí níž bude hodnota získána a uložení do pole funkcí třídy `Printer`,
3. synchronizace získaných statických dat s databází,
4. vložení všech funkcí pro získávání dat dané tiskárny do kalendáře událostí.



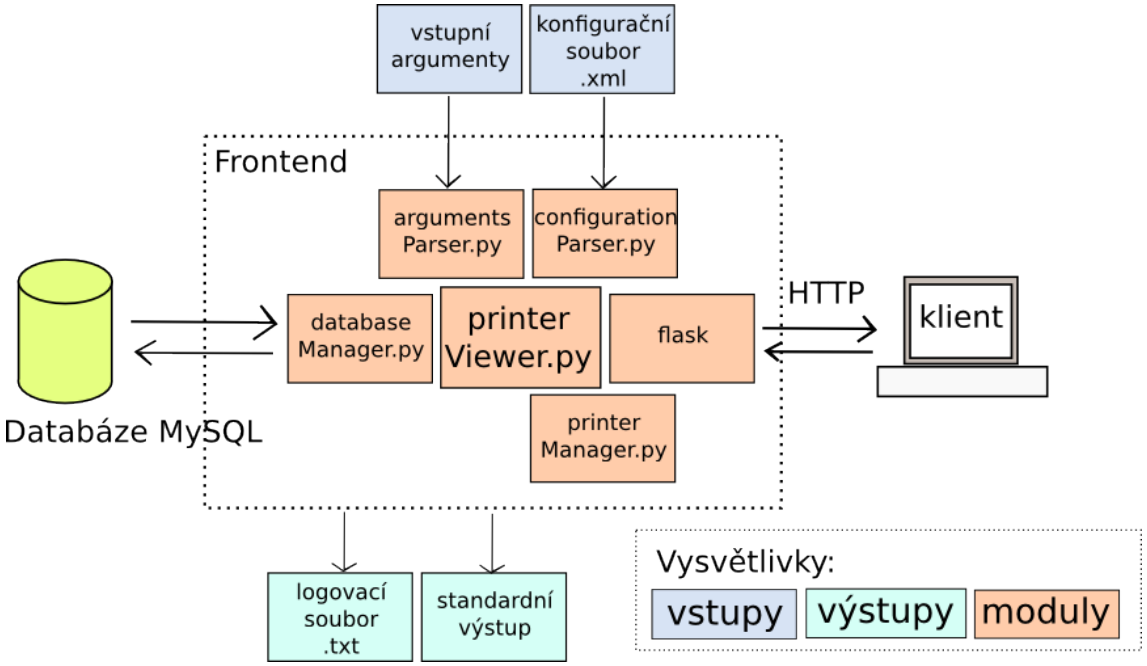
Obrázek 4.2: Činnost vázacího manageru

4.2.8 Moduly pro zjednodušení získávání dat - `SnmpGetter`, `HttpGetter`

Uživatelům, kteří píší své vlastní moduly a třídy pro získávání dat jsou přístupné pomocné moduly (*snmpGetter.py*, *httpGetter.py*), jenž implementují nejčastější operace spojené se získáváním dat. Psaní vlastních tříd je pro ně poté jednodušší. Z důvodu náročnosti bakalářské práce nebyl modul pro HTTP dopsán. Veškerou podporu pro HTTP však aplikace obsahuje.

4.3 Implementace frontendu

Frontendová část je sestavena z modulů. Každý modul je řešen jako třída, jenž se stará o jemu přidělenou část, stejně jako v popisu modulů v backendové části (viz 4.2). Jádro (*printerViewer.py*) je hlavní smyčka programu, ve které jsou tyto moduly instanciovány a volány jejich funkce. Struktura frontendu je vidět na následujícím obrázku 4.3.



Obrázek 4.3: Podrobné programové schéma frontendu

Dle tohoto návrhu jsou moduly volány z hlavní smyčky programu (*printerViewer.py*) v následujícím pořadí:

1. načítání a parsování všech vstupních argumentů (*argumentsParser.py*),
2. načítání a parsování konfiguračního souboru s hlavním nastavením (*configurationParser.py*),
3. vytvoření databázového spojení, validování databázového schématu, popř. vytvoření nového (*databaseManager.py*),
4. vytvoření a spuštění webového serveru (*flask*),
5. vytvoření modulu, jenž slouží jako model v architektuře MVC (*printerManager.py*).

Nyní si popíšeme význam jednotlivých modulů a jejich funkce.

4.3.1 Parser argumentů - `ArgumentsParser`

Parser argumentů načítá veškeré argumenty předané programu při jeho spuštění. Mezi nejhlavnější předané argumenty patří:

- -c [] -> cesta ke konfiguračnímu souboru s nastavením databáze,
- -l [] -> cesta k výstupnímu logovacímu souboru,
- -v -> zapnutí debugovacího módu (ladící výpisy),
- –develop -> zapnutí vývojové verze, která spustí webový server pouze na lokálním rozhraní 127.0.0.1:5000,
- –debug -> zapnutí ladících výpisů ve webovém rozhraní aplikace.

4.3.2 Parser konfigurace - `ConfigurationParser`

Parser konfigurace načítá konfigurační XML soubor, v němž se nacházejí data určující přístup k databázi (přihlašovací údaje, adresa serveru, číslo portu). Data se předávají tímto způsobem stejně jako v backendu kvůli bezpečnosti.

4.3.3 Databázový manager - `DatabaseManager`

Tento modul je shodný s modulem pro backendovou část. Zastává stejnou funkci a obsahuje totožné metody. Tento modul je jediná věc, jenž mají backend a frontend společnou.

4.3.4 Manager dat - `PrinterManager`

Modul sloužící jako spojka mezi webovým rozhraním a databází. V MVC přístupu se jedná o modelovou část. Slouží k načtení informací o tiskárnách a jejich hodnotách. Tyto hodnoty dále upravuje tak, aby mohly být prezentovány webovým rozhraním.

4.3.5 Webový server - `flask`

Jedná se o modul třetí strany, jenž zastává funkci webového serveru. Dalo by se říci, že pracuje pomocí MVC přístupu, kdy modelem je naše třída *PrinterManager*. Webový server běží na portu 5000 nezabezpečeného spojení HTTP. Tento modul je popsán v podkapitole 4.4.3.

4.4 Použité knihovny

Nyní si popíšeme knihovny a utility použité k hlavním účelům aplikace.

4.4.1 Databáze - *pymysql*

Tato knihovna [9] ve verzi 0.7.10 představuje klientskou část v komunikaci s databázovým serverem (daemonem), běžícím buď na lokálním či vzdáleném rozhraní. Daemon primárně naslouchá na portu 3306. K navázání komunikace potřebujeme uživatelské jméno, heslo, jméno databáze, adresu stroje s běžícím daemonem a číslo portu. Jakmile je ustanovena komunikace, lze se databáze dotazovat dvěma typy:

- po vykonání dotazu dostaneme celou odpověď (použito v aplikaci),
- po vykonání dotazu dostaneme pouze jeden záznam a postupnou iterací dostáváme od serveru i zbývající.

Rozhraní dotazů funguje pomocí příkazu query. Tento query příkaz očekává zápis příkazu v MySQL syntaxi, na který jsou následně vrácená data - buď jako asociativní pole, nebo pole obyčejné.

4.4.2 Konfigurace - *lxml*

Tato knihovna [10] ve verzi 3.4.0 se používá pro práci s XML a HTML soubory. V této aplikaci byla použita pro parsování XML konfigurací, pro backendovou i frontendovou část. Výhodou knihovny je její jednoduché API a možnost validovat schéma oproti DTD schématu, jenž bylo v aplikaci využito, aby nebylo potřeba provádět ruční parsování a kontrolu.

4.4.3 Webový server - *flask*

Jedná se o microframework pro python [11] ve verzi 0.12, jenž byl použit jako webový server pro frontendovou část. Po správném nakonfigurování a spuštění je webový server přístupný pod portem 5000 nezabezpečeného spojení HTTP. Použití tohoto webového serveru je velmi jednoduché. Umožňuje veškeré funkce jako konkurenční řešení: šablonování, routování, ajaxové požadavky, debugovací mód, aj.

Webové stránky

Pro reprezentaci webových stránek je použit šablonovací systém *jinja2* ve verzi 2.9.5. Vytvořeny byly následující šablony:

- *template.html* - šablona, jenž obsahuje hlavičku, tělo a patičku, do kterých jsou vloženy hodnoty definované v dalších stránkách. Díky tomu je vzhled rozhraní pro všechny stránky stejný,

- *index.html* - stránka, jenž vypisuje skupiny a jejich metainformace,
- *group.html* - stránka, jenž vypisuje tiskárny ve zvolené skupině a jejich metainformace,
- *printer.html* - stránka, jenž vypisuje veškeré informace k dané tiskárně (spotřební materiály a jejich hodnoty, varování, statická data, ...),
- *404.html* - stránka, jenž se zobrazí při HTTP erroru 404,
- *500.html* - stránka, jenž se zobrazí při HTTP erroru 500.

Ukázky výsledných webových stránek lze nalézt v příloze C.
K řazení tabulek byl využit jquery tablesorter [12] ve verzi 2.0.

4.4.4 Další utility

Mezi další použité utility patří utility shellu, které byly externě volány. Použity byly:

- *snmpget, snmpgetnext* - pro získávání hodnot z tiskáren pomocí protokolu SNMP,
- *ping, ping6* - pro ověřování, zda je tiskárna dostupná a probouzení pomocí ICMP paketů,
- *netcat* (nc) - pro probouzení tiskáren pomocí otevření TCP spojení na jejich webové servery,
- *echo* - pro předání vstupu pro netcat, jenž má být poslán v paketu jako data.

Kapitola 5

Nasazení v síti

Aplikace byla nasazena ve fakultní síti FIT VUT na serveru s operačním systémem FreeBSD. Jako vstup posloužily tiskárny, jejichž výčet je uveden v tabulce 2.1 a byly s nimi testovány konkurenční aplikace.

5.1 Postup

Popis pro zprovoznění aplikace je popsán v README souboru, který se nachází ve složce *PriMoVi/* (viz příloha B).
Pro monitorování tiskáren jsem provedl následující postup (stručně):

1. spuštění databázového daemona a vytvoření nové databáze, jenž neobsahovala žádná data,
2. pro všechny tiskárny které byly sledovány - automatické vygenerování konfigurace pomocí skriptu popsaného v podkapitole 4.2.2 (automatické generování na základě IP adresy tiskárny, či doménového jména a jejího unikátního (vymyšleného) názvu pro identifikaci v databázi),
3. přidání tiskáren (jejich vygenerovaných konfigurací) do hlavní konfigurace (viz podkapitola 3.2.2) a dopsání sekce `<general>` v hlavní konfiguraci, jenž obsahuje přihlašovací údaje k databázi a další informace (nastaveny defaultně, není třeba dopisovat), jako například interval po jakém se bude zjišťování dat opakovat, nastavení SNMP, ...,
4. vytvoření konfiguračního souboru pro frontendovou část (zadáním pouze přihlašovacích údajů k databázi),
5. spuštění backendové části a po její úspěšné incializaci i spuštění části frontendové.

5.2 Výsledky

Byly rozpoznány veškeré tiskárny a získány všechny spotřební materiály kromě tiskárny Canon, jenž nenabízela žádné spotřební materiály z *Printer MIB* databáze. Moduly této tiskárny mohou být implementovány pomocí přidání HTTP modulů. Pro srovnání s ostatními aplikacemi jsem vyhotovil stejnou tabulku jako při testování konkurenčních aplikací.

Vyplňovány však nebyly sloupce: status papíru, počet stran a status, jenž jsou momentálně nedostupné a lze snadno implementovat přidáním dalších modulů pro získávání dat.

Tiskárna	Spotřeba černého a barevného toneru	Varování	Run time
Xerox	procenta	ano	ano
Konica Minolta	procenta	ano	ano
Brother	procenta	ano	ano
Samsung	procenta	ano	ano
HP	procenta	ano	ano
Canon	nedostupné	ne	ano
OKI	procenta	ano	ano
Epson	procenta	ano	ano

Tabulka 5.1: Výsledky testování tiskáren výslednou aplikací

Kapitola 6

Závěr

6.1 Rozšíření do budoucna

Aplikaci je možno dále rozšiřovat. Do aplikace lze přidávat další SNMP či HTTP moduly pro získávání dalších dat potřebných k monitorování tiskáren. Tyto moduly může správce sítě psát ručně, nebo je vyměňovat se správci ostatními. Přidáním modulů bude aplikace získávat stále více informací, které jsou pro jejich správce důležité. Pro základní údaje jsou však přiložené moduly dostačující.
Časem může také vzniknout databáze dostupných modulů, které by byly stáhnutelné z internetu, nebo dostupné přímo v balíčku aplikace. Spolu s moduly by mohla obsahovat i stáhnutelné předvyplněné konfigurace pro přesně dané typy tiskáren.
Webové rozhraní aplikace lze upravit tak, aby zobrazovalo další data, jako například grafické znázornění vývoje hodnot spotřebních materiálů v čase. Přidání této úpravy je velice jednoduché.
Pro hlubší analýzu se dá z časových řad vypočítat rychlost úbytku spotřebních materiálů a odvodit dobu dalšího výpadku. Správce sítě by tak mohl dopředu objednat tonery, které s největší pravděpodobností dojdou.
Lze také snadno dopsat modul *HttpGetter*, který by obsahoval zapouzdřené metody k získávání dat pomocí protokolu HTTP. Samotnou podporu získávání dat pomocí protokolu HTTP aplikace obsahuje.
Pro upozornění správců by se dala dopsat funkcionalita notifikace e-mailem při překročení hraničních hodnot.

6.2 Zhodnocení práce

Cílem této bakalářské práce bylo prostudovat existující systémy pro monitorování spotřebních materiálů síťových tiskáren a navrhnout řešení tak, aby obsahovalo podporu pro tiskárny různých výrobců. Bylo zjištěno, že všechny testované aplikace neuměly získávat data ze všech tiskáren námi testovaných typů a byl navrhnut nový přístup k získávání dat. Vzhledem ke komplexnosti aplikace se návrh potýkal s mnohými problémy jako například s výběrem databáze, či jakým způsobem jednotně získávat data z tiskáren.
Po návrhu byla aplikace implementována s použitím skriptovacího jazyku Python a otestována na fakultní síti. Ukázalo se, že aplikace zvládla získat a přehledně zobrazit veškeré hodnoty potřebné pro její monitorování. Vzhledem k časové tísni nebyly implementovány

moduly pro získávání dat metodou HTTP.
Aplikace je plně funkční komplexní řešení, které najde uplatnění u správců sítí větších organizací, jenž mají v síti tiskárny různých výrobců a potřebují jejich jednotné monitorování. Aplikace je dále snadno rozšiřitelná, v čemž by mohla pokračovat navazující diplomová práce.

Literatura

[1] *BRAdmin Professional: Support & Downloads*. Brother [online]. Brother Industries [cit. 2016-11-13]. Dostupné z: http://support.brother.com/g/s/id/bradmin_pro/en/index.html.

[2] BURROWS, Steven. *Laser printer secrets* [online]. West Topsham, Vt: BlackLightning, 1996 [cit. 2016-11-16]. ISBN 18-816-7605-6. Dostupné z: http://www.poota.com/lpbook/00-toc.html.

[3] Technická zpráva, case, J. D.; Fedor, M.; Schoffstall, M. L.; aj.: *Simple Network Management Protocol (SNMP)*. RFC 1157, RFC Editor, Květen 1990. Dostupné z: http://www.rfc-editor.org/rfc/rfc1157.txt.

[4] *An Introduction to SNMP (Simple Network Management Protocol)* [online]. In: . DigitalOcean, c2017 [cit. 2017-01-20]. Dostupné z: https://www.digitalocean.com/community/tutorials/an-introduction-to-snmp-simple-network-management-protocol.

[5] Technická zpráva, bergman, R.; Lewis, H.; McDonald, I.: *Printer MIB v2*. RFC 3805, RFC Editor, červen 2004. Dostupné z: https://tools.ietf.org/html/rfc3805.

[6] Technická zpráva, mcCloghrie, K.; Rose, M. T.: *Management Information Base for Network Management of TCP/IP-based internets:MIB-II*. RFC 1213, RFC Editor, březen 1991. Dostupné z: http://www.rfc-editor.org/rfc/rfc1213.txt.

[7] Technická zpráva, fielding, R. T.; Gettys, J.; Mogul, J. C.; aj.: *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616, RFC Editor, červen 1999. Dostupné z: http://www.rfc-editor.org/rfc/rfc2616.txt.

[8] ZAHARIEV, Ivan. *C++ vs. Python vs. Perl vs. PHP performance benchmark*. In: Famzah's blog [online]. [cit. 2017-01-16]. Dostupné z: https://blog.famzah.net/2016/02/09/cpp-vs-python-vs-perl-vs-php-performance-benchmark-2016/.

[9] *PyMySQL's documentation* [online]. c2016 [cit. 2017-01-20]. Dostupné z: https://pymysql.readthedocs.io/en/latest/.

[10] *Lxml - XML and HTML with Python* [online]. 2017 [cit. 2017-02-02]. Dostupné z: http://lxml.de/.

[11] *Flask - python microFramework* [online]. c2010-2017 [cit. 2017-04-02]. Dostupné z: http://flask.pocoo.org/.

[12] *Tablesorter: jQuery plugin: Tablesorter 2.0* [online]. [cit. 2017-04-28]. Dostupné z: http://tablesorter.com/docs/.

[13] Xerox Corporation: *Xerox Device Agent (XDA) Lite Security and Evaluation Guide*. Čtvrté vydání, červen 2013.

[14] Xerox Corporation: *Xerox Device Agent (XDA) Lite User Guide*. Čtvrté vydání, červen 2013.

Přílohy

Příloha A

Obsah přiloženého paměťového média

Přiložené paměťové médium obsahuje:

- ./PriMoVi/ - adresář s projektem aplikace,
- ./doc/ - adresář se zdrojovými kódy této technické zprávy,
- ./technickaZprava.pdf - tuto technickou zprávu ve formátu PDF.

Příloha B

Souborová struktura projektu aplikace PriMoVi

Aplikace PriMoVi obsahuje následující soubory a složky:

```
PriMoVi/                    # hlavní složka s projektem
  ./README                  # soubor obsahující základní instrukce pro zprovoznění projektu
  ./requirements.txt        # požadavky aplikace
  ./printerMonitor/         # složka backendové části projektu
    ./printerMonitor.py     # spouštěcí soubor backendové části
    ./README                # soubor obsahující instrukce pro zprovoznění backendové části
    ./config/               # složka pro konfigurační soubory
      ./configDescription.xml  # komentovaný konfigurační soubor
      ./configTemplate.xml     # ukázkový konfigurační soubor
    ./logs/                 # složka pro výstupní logovací soubor
    ./utils/                # složka obsahující pomocné utility
      ./configurationGetter.py  # pomocný skript pro získávání konfigurace tiskáren
    ./src/                  # složka se zdrojovými soubory (moduly) k backendové části
      ./___init___.py       # incializační soubor tvořící balíček v pythonu
      ./argumentsParser.py
      ./configurationParser.py
      ./modulesParser.py
      ./databaseManager.py
      ./printer.py
      ./printerBinder.py
      ./eventsCalendar.py
      ./argumentsParser.py
      ./snmpGetter.py
      ./httpGetter.py
      ./printersModules/    # složka obsahující moduly pro získávání dat z tiskáren
        ./printerBrother.py # modul získávající data pomocí Brother MIB
        ./printerMIB.py     # modul získávající data pomocí Printer MIB
        ./printerSystem.py  # modul získávající data pomocí System MIB
```

```
./printerViewer/          # složka frontendové části projektu
  ./printerViewer.py      # spouštěcí soubor frontendové části
  ./README                # soubor obsahující instrukce pro zprovoznění frontendové části
  ./config/               # složka pro konfigurační soubory
    ./configDescription.xml  # komentovaný konfigurační soubor
    ./configTemplate.xml     # ukázkový konfigurační soubor
  ./logs/                 # složka pro výstupní logovací soubor
  ./static/               # složka obsahující statické soubory k webové aplikaci
    ./style.css
    ./jquery-latest.js
    ./jquery.tablesorter.js
    ./home-icon.png
    ./sort_down.png
    ./sort_up.png
    ./sort_neutral.png
  ./templates/            # složka obsahující šablony webových stránek
    ./404.html
    ./500.html
    ./index.html
    ./group.html
    ./printer.html
    ./template.html
  ./src                   # složka se zdrojovými soubory (moduly) k backendové části
    ./___init___.py       # incializační soubor tvořící balíček v pythonu
    ./argumentsParser.py
    ./configurationParser.py
    ./databaseManager.py
    ./prinerManager.py
```

Příloha C

Ukázky webového výstupu aplikace

Overview of printer groups

Group	Active	Inactive	Warnings	< 20 %	20-40 %	40-60 %	60-80 %	> 80 %
All printers	8	0	7	13	7	10	6	18
Epson	1	0	0	1	0	2	1	0
Samsung	1	0	2	2	1	3	0	4
HP	1	0	0	1	0	3	0	0
Canon	1	0	0	0	0	0	0	0
Xerox	1	0	0	1	1	1	1	5
Brother	1	0	1	1	2	0	1	2
OKI	1	0	0	5	1	1	2	1
Konica Minolta	1	0	4	2	2	0	1	6
No group	0	0	0	0	0	0	0	0

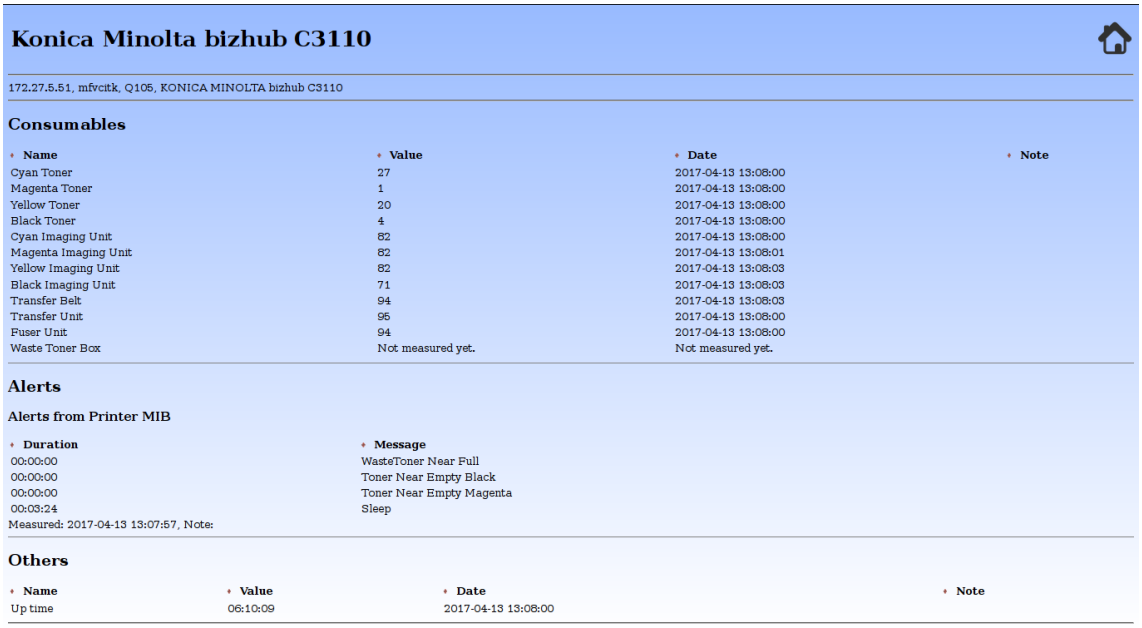
Obrázek C.1: Hlavní stránka s přehledem skupin

Group All printers

Active printers

Dns	Vendor	Type	Group	Warnings	< 20 %	20-40 %	40-60 %	60-80 %	> 80 %
mfvcitk	Konica Minolta	bizhub C3110	Konica Minolta	4	2	2	0	1	6
bxrc	Xerox	Phaser 6300DN	Xerox	0	1	1	1	1	5
mflib	Brother	NC-6900h	Brother	1	1	2	0	1	2
mfs108	HP	Laser Jet CM2320fxi MFP	HP	0	1	0	3	0	0
mfpo	Samsung	XLC-6220 Series	Samsung	2	2	1	3	0	4
mfs110	Canon	MF4360-4390	Canon	0	0	0	0	0	0
mfvcit	OKI	OkiLan 8600e	OKI	0	5	1	1	2	1
mfs205	Epson	Unknown	Epson	0	1	0	2	1	0

Obrázek C.2: Stránka dané skupiny s přehledem tiskáren



Konica Minolta bizhub C3110

172.27.5.51, mfvcitk, Q105, KONICA MINOLTA bizhub C3110

Consumables

Name	Value	Date	Note
Cyan Toner	27	2017-04-13 13:08:00	
Magenta Toner	1	2017-04-13 13:08:00	
Yellow Toner	20	2017-04-13 13:08:00	
Black Toner	4	2017-04-13 13:08:00	
Cyan Imaging Unit	82	2017-04-13 13:08:00	
Magenta Imaging Unit	82	2017-04-13 13:08:01	
Yellow Imaging Unit	82	2017-04-13 13:08:03	
Black Imaging Unit	71	2017-04-13 13:08:03	
Transfer Belt	94	2017-04-13 13:08:03	
Transfer Unit	95	2017-04-13 13:08:00	
Fuser Unit	94	2017-04-13 13:08:00	
Waste Toner Box	Not measured yet.	Not measured yet.	

Alerts

Alerts from Printer MIB

Duration	Message
00:00:00	WasteToner Near Full
00:00:00	Toner Near Empty Black
00:00:00	Toner Near Empty Magenta
00:03:24	Sleep

Measured: 2017-04-13 13:07:57, Note:

Others

Name	Value	Date	Note
Up time	06:10:09	2017-04-13 13:08:00	

Obrázek C.3: Stránka dané tiskárny