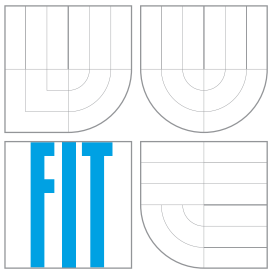
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

WEBOVÉ KONFIGURAČNÉ ROZHRANIA PRE SIEŤOVÉ ZARIADENIA

WEB CONFIGURATION INTERFACE FOR NETWORK DEVICES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MAREK ŽIŽLAVSKÝ
AUTHOR

VEDOUCÍ PRÁCE — Ing. TOMÁŠ MARTÍNEK
SUPERVISOR

BRNO 2007

Zadanie bakalárskej práce

1. Seznamte se s HW a SW architekturou síťových zařízení vytvořených v rámci projektu Liberouter.
2. Analyzujte možnosti a potřeby konfigurace těchto vysokorychlostních síťových zařízení. Zaměřte se na oblast konfigurace přes webové rozhraní.
3. Navrhněte webové konfigurační rozhraní pro pasivní monitorovací sondu FlowMon. K ovládání sondy využijte protokol NETCONF.
4. Proveďte realizaci navrženého webového rozhraní a ověřte jeho funkcionalitu se sondami FlowMon.

Licenčná zmluva

Licenčná zmluva je uložená v archíve Fakulty informačných technológií Vysokého učenia technického v Brne.

Abstrakt

Predkladaná práca sa zaoberá návrhom obecného konfiguračného systému na vzdialenú konfiguráciu sieťových zariadení s využitím protokolu NETCONF, a návrhom a implementáciou webového konfiguračného rozhrania pre monitorovaciu sondu FlowMon. Práca naväzuje na existujúce programové vybavenie vytvorené vrámci projektu Liberouter. Implementačná časť sa zaoberá vytvorením webového konfiguračného rozhrania, podporných knižníc na prácu s konfiguráciou a knižnicou na prácu s GUI v jazyku PHP. Implementované konfiguračné rozhranie pre sondu FlowMon, zároveň slúži ako funkčný prototyp konfiguračného rozhrania založeného na koncepcií obecného konfiguračného systému navrhnutého vrámci tejto bakalárskej práce.

Klíčová slova

Konfigurácia, sieťové zariadenia, webové konfiguračné rozhranie, Liberouter, FlowMon, NETCONF

Abstract

Main goal of this bachelor work is to design a universal configuration system for remote network device configuration based on the NETCONF protocol and to develop a web configuration interface for the FlowMon, monitoring probe. This work uses existing software tools developed by the Liberouter project. Main objective of the implementation part of this work is to develop a web configuration interface, supporting class libraries for configuration management and GUI creation in PHP. Implemented web configuration interface for the FlowMon probe is a fully-functional prototype of a configuration interface based on concepts of the universal configuration system designed in this bachelor work.

Keywords

Configuration, network devices, web configuration interface, Liberouter, FlowMon, NETCONF

Citace

Marek Žižlavský: Webové konfiguračné rozhrania pre sieťové zariadenia, bakalářská práce, Brno, FIT VUT v Brně, 2007

Webové konfiguračné rozhrania pre sieťové zariadenia

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pana Ing. Tomáša Martínka

.....
Marek Žižlavský
15. května 2007

Poděkování

Ďakujem svojemu vedúcemu Ing. Tomášovi Martínkovi za pomoc a cenné rady, ktoré mi pomohli pri smerovaní práce. Ďakujem aj kolektívu projektu Liberouter za možnosť realizovať prácu vrámci projektu a za informačnú a technickú podporu.

Obsah

Kapitola 1

Úvod

Každá firma vyvíjajúca sieťové zariadenia musí riešiť otázku ich konfigurácie. Pre menej skúsených užívateľov je dôležité, aby mohli bez problémov nastaviť základné parametre zariadenia. Ideálnym riešením je webové konfiguračné rozhranie, pretože súčasné webové riešenia založené na jazyku HTML umožňujú vytvárať užívateľsky príťažlivé grafické užívateľské rozhrania, prístupné aj pre vzdialeného používateľa. Zároveň sú tieto webové aplikácie platformovo nezávislé, čo umožňuje ich použitie s ľubovoľným operačným systémom, na ktorý je možné nainštalovať webový prehliadač. To umožňuje administrátorom konfiguráciu pomocou PDA, inteligentných mobilných telefónov a pod.

Štandardy v oblasti konfigurácie pomocou webového rozhrania zatiaľ neboli definované a preto má každý výrobca sieťových zariadení svoje vlastné proprietárne riešenie. Táto bakalárska práca sa snaží navrhnúť obecne použiteľné konfiguračné riešenie nezávislé na zariadení. Pri návrhu takéhoto riešenia je potrebné vyriešiť najmä spôsob dátovej reprezentácie konfigurácie a spôsob komunikácie konfigurovaného zariadenia a webového rozhrania. Dôležitá je aj otázka bezpečnosti, preto je potrebné vyriešiť možnosť overenia identity užívateľa a zabezpečiť integritu a utajenie prenášaných konfiguračných dát, pretože tie môžu často obsahovať aj citlivé informácie.

Analýza požiadaviek kladených na takýto obecný konfiguračný systém je spracovaná v kapitole 2. Ako prípadová štúdia nasadenia navrhovaného riešenia bola zvolená implementácia webového konfiguračného rozhrania pre sondu FlowMon, vyvíjanú v rámci projektu Liberouter. Preto druhá kapitola obsahuje aj základný popis sieťových zariadení vyvíjaných v rámci projektu ako aj charakteristiku samotného projektu Liberouter. Podrobnejšia charakteristika pasívnej monitorovacej sondy FlowMon sa nachádza v podkapitole 2.3.

Základom navrhovaného riešenia je využitie protokolu NETCONF, ktorý slúži na správu konfiguračných dát. Charakteristika protokolu, a popis jeho architektúry sa nachádza v kapitole 2.4. Detailné diagramy procesov, ktoré prebiehajú pri použití vybraných operácií protokolu NETCONF používaných webovým rozhraním sú uvedené v podkapitole 3.2 kapitoly 3.

Kapitola 3 sa zaoberá návrhom obecného konfiguračného systému, popisuje navrhovanú architektúru, jej jednotlivé vrstvy a funkcie, ktoré zabezpečujú. Ďalej sa venuje popisu návrhu webového konfiguračného rozhrania pre sondu *FlowMon*. Popisuje jeho začlenenie v rámci architektúry celého konfiguračného systému.

Počas vývoja webového rozhrania pre sondu FlowMon bolo vytvorených viacero znovu použiteľných tried, ktoré boli umiestnené do samostatných knižníc. Ich detailným popisom sa zaoberá kapitola 4. Postupne opisuje možnosti a funkcie knižnice *libphpconf*, do ktorej

boli zaradené triedy umožňujúce použitie obecného konfiguračného systému v jazyku PHP, knižnice *libphpctrl*, ktorá obsahuje triedy na podporu objektovo orientovaného prístupu k práci s GUI ako aj popis implementácie samotného webového konfiguračného rozhrania.

Prezentované riešenie pre sondu FlowMon slúži ako plne funkčný prototyp webového konfiguračného rozhrania postaveného na základoch navrhovaného obecného konfiguračného systému. Na jeho základe bude možné vytvoriť konfiguračné rozhrania aj pre ostatné zariadenia vyvíjané v rámci projektu Liberouter. Navrhnutý konfiguračný systém však nie je na využitie iba v rámci projektu nijak viazaný a preto môže slúžiť minimálne ako inšpirácia aj pre ostatných výrobcov.

V záverečnej kapitole je celá práca zhrnutá, je zhodnotené naplnenie cieľov vyplývajúcich zo zadania a je načrtnutá vízia smerovania do budúcnosti.

Kapitola 2

Analýza

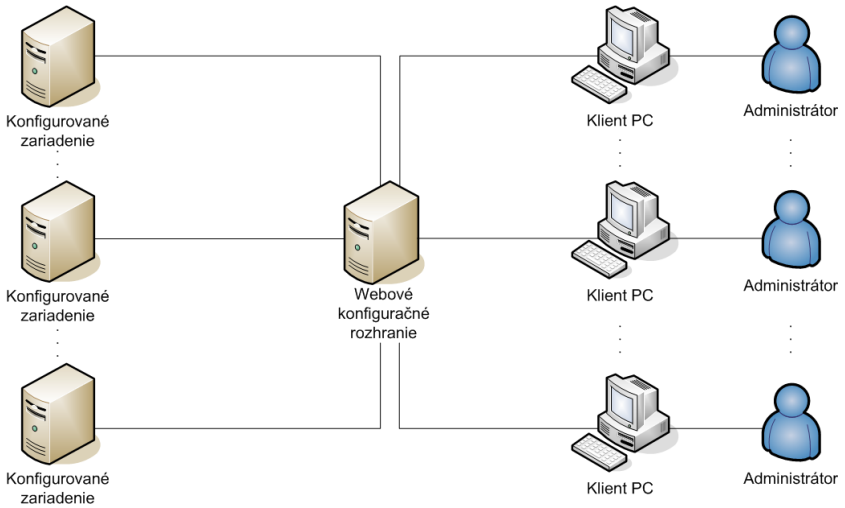
Každé sieťové zariadenie, má svoje špecifické požiadavky na konfiguráciu. To je spôsobené tým, že každé zariadenie má svoje vlastné parametre, ktoré možno nastavovať. Táto práca sa preto snaží navrhnúť obecný konfiguračný systém, ktorý bude nezávislý na konkrétnom zariadení.

2.1 Požiadavky

Konfiguračný systém s ohľadom na možnosť konfigurácie pomocou webového rozhrania musí spĺňať nasledovné požiadavky:

- možnosť konfigurácie na diaľku,
- zabezpečenie integrity a utajenia prenášaných dát, overenia identity užívateľa,
- možnosť konfigurácie viacerých zariadení z jedného miesta viacerými užívateľmi súčasne.

Obecný prípad, keď viacerí sieťoví administrátori súčasne konfigurujú pomocou webového rozhrania viacero sieťových zariadení je zobrazený na obrázku 2.1.



Obrázok 2.1: Schéma nasadenia webového konfiguračného rozhrania

Pri návrhu bude potrebné vyriešiť niekoľko problémov ako je:

- dátová reprezentácia konfigurácie,
- komunikácia s konfigurovaným zariadením,
- nástroj na editáciu konfigurácie užívateľom,
- bezpečnosť.

Konfigurácia pomocou webového konfiguračného rozhrania má na konfiguračný systém ďalšie nároky. S princípu sa jedná o architektúru klient - server. Serverová časť je tvorená webovým serverom, klientska časť je tvorená webovým prehliadačom. Presunutie webového serveru a webového konfiguračného rozhrania na samostatný server umožňuje konfiguráciu viacerých zariadení súčasne. Riešenia, ktoré integrujú webový server a konfiguračné rozhranie do jedného sieťového zariadenia takúto možnosť z princípu architektúry neumožňujú. Presunutie webového rozhrania na dedikovaný server je výhodné aj z pohľadu stability, výkonu a bezpečnosti daného riešenia.

2.2 Konfiguračný systém pre projekt Liberouter

Kapitola sa zaoberá analýzou použitia obecného konfiguračného systému navrhnutého v rámci tejto bakalárskej práce na konfiguráciu zariadení vyvinutých v rámci projektu Liberouter. Predkladá základné informácie o projekte, hardwarovej platforme, ktorú používa ako aj o jednotlivých zariadeniach vyvíjaných v rámci projektu Liberouter.

2.2.1 Liberouter

Pôvodným cieľom projektu Liberouter bol vývoj multigigabitového IPv4, IPv6 smerovača postaveného na architektúre PC a otvorenom dizajne so softwérom a firmwérom s otvoreným zdrojovým kódom (Open Source Software). Na urýchlenie preposielania a filtrovania paketov bol vyvinutý špeciálny hardwarový akcelerátor COMBO, ktorý používa technológiu FPGA [1]. Vďaka otvorenému dizajnu si COMBO našlo uplatnenie v rôznych zaujímavých aplikáciách v oblasti počítačových sietí.

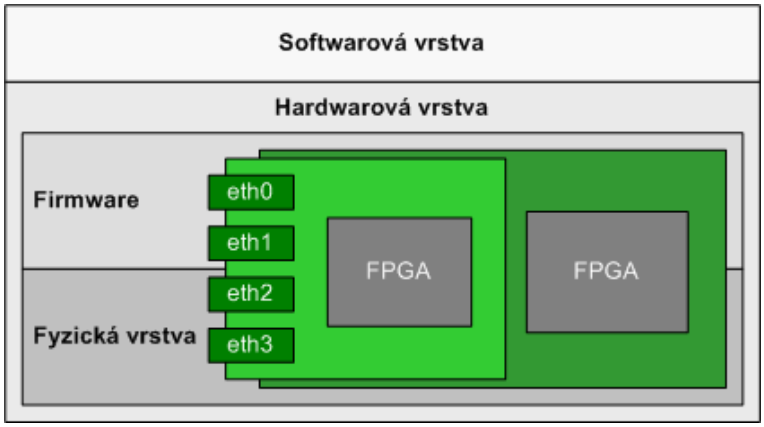
2.2.2 Hardware

Ako už bolo spomenuté vysokorýchlostné sieťové zariadenia vyvíjané v rámci projektu Liberouter používajú platformu PC a hardwarový akcelerátor COMBO. Jednotlivé vrstvy architektúry sieťových zariadení postavených na platforme COMBO sú zobrazené na obrázku 2.2.

Fyzická vrstva: Sieťové zariadenia sú postavená na hardwarovej platforme COMBO [2], ktorá je osadená programovateľným hradlovým poľom *FPGA*, čo ju činí univerzálnou. Spojenie so sieťou zabezpečuje rozširujúca karta so sieťovým rozhraním a prídavným hradlovým poľom.

[1] Field-Programmable Gate Arrays

[2] Detaily hardwarovej špecifikácie jednotlivých verzií kariet sú dostupné v Dodatku A.



Obrázok 2.2: Vrstvy architektúry sieťových zariadení projektu Liberouter

Firmware: Táto vrstva zabezpečuje funkcionalitu celého sieťového zariadenia. Obyčajne obsahuje výkonné jadro systému, ktoré spracováva dáta prechádzajúce sieťou. Jadro môže obsahovať viacero paralelne pracujúcich jednotiek, ktoré dokážu analyzovať, prípadne aj editovať hlavičky paketov a ich obsah (payload). Jednotlivé komponenty sú implementované v jazyku VHDL a Handel-C.

Softwarová vrstva: Zabezpečuje prepojenie nižších vrstiev s jadrom operačného systému a poskytuje užívateľské nástroje na prácu so zariadením.

2.2.3 Projekty

V súčasnosti je v rámci projektu Liberouter realizovaných niekoľko výskumných projektov:

Liberouter: 4-portový Gigabitový dual-stack router s podporou protokolov IPv4 a IPv6. Jedná sa o pôvodný projekt realizovaný v rámci projektu Liberouter.

Scampi: Pasívny sieťový monitorovací adaptér navrhnutý pre technológie 1 Gbps a 10 Gbps.

NIC: 4-portová sieťová karta s podporou 1 Gbps technológie. Tento dizajn sa pre svoju relatívnu jednoduchosť používa ako testovací dizajn po oživení nových kariet COMBO a pri vývoji nových verzií jednotlivých spracovávacích jednotiek v súvislosti s prechodom na platformu NetCOPE.

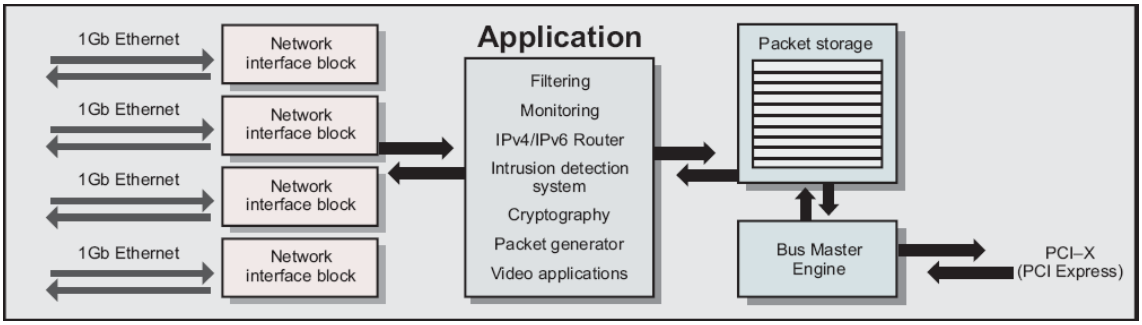
NIFIC: 4-portová sieťová karta s podporou 1 Gbps technológie a hardware podporou filtrovania a preposielania paketov.

FlowMon: Pasívna sonda na monitorovanie tokov podporujúca technológie 1 Gbps a 10 Gbps. Detailný popis sondy sa nachádza v kapitole 2.3.

Traffic Scanner: Zariadenie na detekciu prienikov do siete. Traffic Scanner slúži ako hardwarový akcelerátor IDS [3] programu Snort. Funguje na princípe vyhľadávania jednoduchých reťazcov určených Snort pravidlami, avšak nedisponuje podporou vyhľadávania regulárnych výrazov.

[3] Intrusion Detection System

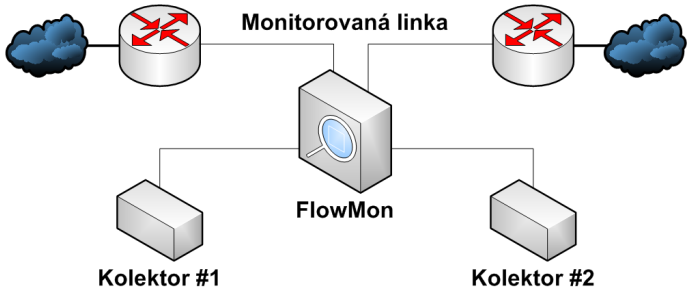
NetCOPE: Platforma určená na rýchly vývoj sieťových aplikácií využívajúcich karty COMBO. NetCOPE definuje obecný systém zberníc a množstvo rozličných spracovávacích jednotiek *IP Cores*. Zároveň rieši komunikáciu so systémom pomocou systémovej zbernice. Vytvára tým jednotné prostredie pre aplikácie (FlowMon, Traffic Scanner, NIFIC...), ktoré sú potom relatívne nezávislé na konkrétnej hardwarovej platforme, t.j. na verzii COMBO karty. Schéma architektúry je zobrazená na obrázku 2.3.



Obrázok 2.3: Architektúra NetCOPE

2.3 Sonda FlowMon

FlowMon je pasívna autonómna monitorovacia sonda, ktorá zbiera informácie o dátach putujúcich sieťou. Inšpiráciou pri návrhu sondy bol systém NetFlow implementovaný firmou Cisco. NetFlow poskytuje agregované informácie o tokoch protokolu IP, ktoré môžu byť využité pri manažovaní existujúcich a plánovaní nových sieťových topológií, dokáže detegovať útoky typu DoS [4] a taktiež môže byť využitý na zúčtovanie množstva prenesených dát. Schéma nasadenia sondy FlowMon pri monitorovaní linky medzi dvoma routermi je na obrázku 2.4.



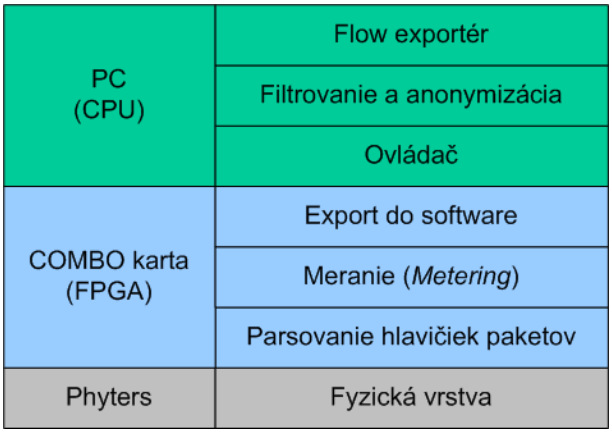
Obrázok 2.4: Sonda FlowMon nasadená na linke, odosielajúca dáta na kolektor

Sonda FlowMon po vložení do linky pracuje ako rozdeľovač (*T-splitter*) vďaka čomu je vložená sonda pre dáta prechádzajúce sieťou plne transparentná. IP toky sú monitorované v hardwarovom akcelerátore COMBO pomocou čipu FPGA a sú ukladané do externej

[4] Denial of Service - odoprenie služby, najčastejšie formou zahltenia nelegitímnymi požiadavkami na službu.

pamäti na karte. Exspirované toky sú následne odosielané na spracovanie do softwarovej vrstvy tvorenej aplikáciu bežiacou v užívateľskom režime operačného systému. Tá spracované toky následne odosiela na kolektor (cez vyhradené sieťové rozhranie). Schéma architektúry sondy FlowMon je zobrazená na obrázku 2.5.

Kolektor môže byť tvorený samostatným počítačom PC, na ktorom beží špecializovaný software. Príkladom softwarových kolektorov sú aplikácie NFDUMP s webovým rozhraním NfSen, FTAS, Ntop. Užívateľské rozhrania kolektorov podporujú zobrazenie informácií o tokoch vo forme prehľadných grafov, dokážu uchovávať históriu zozbieraných tokov a vyhľadávať v nej toky podľa zadaných kritérií.



Obrázok 2.5: Vrstvy architektúry sondy FlowMon

Fyzická vrstva zodpovedá L1 vrstve modelu ISO/OSI.

Parsovanie hlavičiek paketov extrahuje dôležité informácie z paketov. Spracovanie prebieha na L2 a L3 vrstve modelu ISO/OSI.

Meranie (*metering*) zhromažďuje informácie o IP adresách, číslach portov, protokoloch, počíta počet prenesených bytov po linke a počet prenesených paketov, zaznamenáva časové značky a ďalšie údaje z hlavičiek paketov.

Ovládač poskytuje rozhranie na nahratie firmware, nastavenie parametrov merania a umožňuje transport zozbieraných záznamov o tokoch do softwarovej vrstvy.

Anonymizácia modifikuje IP adresy v záznamoch o tokoch aby ochránila súkromie užívateľov.

Filter definuje ktoré záznamy o tokoch budú odoslané na daný kolektor na základe splnenia užívateľom definovaných filtračných pravidiel.

Flow Exportér zaobaľuje záznam o toku do formy NetFlow v5 alebo NetFlow v9 paketov a tie následne odosiela na kolektor.

2.4 NETCONF

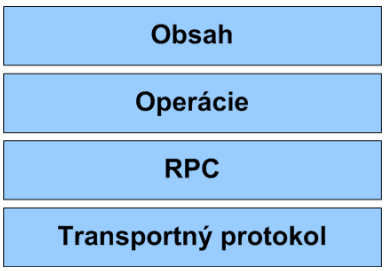
NETCONF je protokol na správu sieťových zariadení vyvinutý v rámci IETF pracovnou skupinou NETCONF Working Group. Bol publikovaný ako RFC 4741.

Protokol NETCONF poskytuje mechanizmy na inštaláciu (nahrávanie), manipuláciu (editáciu) a mazanie konfigurácie sieťových prvkov. Taktiež poskytuje niekoľko funkcií určených na monitorovanie. Na kódovanie konfiguračných dát ako aj samotných protokolových správ používa jazyk XML. Operácie protokolu NETCONF sú vykonávané nad vrstvou protokolu RPC (Remote Procedure Call), ktorý slúži na vzdialené volanie procedúr.
NETCONF môže na prenos správ využiť rôzne transportné protokoly:

- **SSH** - RFC 4742
- **SOAP** - RFC 4743
- **BEEP** - RFC 4744

2.4.1 Architektúra

Koncepčne je možné NETCONF rozdeliť do štyroch vrstiev (viď. obrázok 2.6):



Obrázok 2.6: Vrstvy protokolu NETCONF

Transportný protokol

Najnižšia vrstva transportného protokolu poskytuje spojenie medzi konfigurovaným agentom a manažérom (môžu bežať na rôznych zariadeniach). Táto vrstva môže byť tvorená ľubovoľným protokolom spĺňajúcim požiadavky protokolu NETCONF:

- spoľahlivý prenos dát,
- zabezpečenie integrity dát a ich utajenie,
- služby autentizácie.

Najvýhodnejším protokolom pre transportnú vrstvu sa zdá byť protokol SSH a preto sa s ním ďalej v návrhu konfiguračného systému počíta.

Vzdialené volanie procedúr (RPC)

Protokol NETCONF využíva iba jednoduchý mechanizmus RPC. Manažér vytvorí požiadavok na vykonanie operácie, odošle ho agentovi, ktorý sa následne pokúsi vykonať požadovanú operáciu. RPC model protokolu NETCONF definuje prvky:

- požiadavok (rpc)

- odpoveď (rpc-reply)
- negatívna odpoveď (rpc-error)
- pozitívna odpoveď (ok)

Operácie

Protokol NETCONF definuje sadu operácií určených na konfiguráciu zariadení. Túto sadu je možné v prípade potreby rozšíriť o ďalšie operácie v závislosti na type zariadenia. Základná operácie protokolu NETCONF sú:

- **<get>**
  Operácia získa konfiguráciu running, doplnenú o stavové informácie.
- **<get-config>**
  Operácia získa ľubovoľnú konfiguráciu (startup, running, candidate).
- **<edit-config>**
  Operácia umožňuje editáciu vybranej časti konfiguračných dát.
- **<copy-config>**
  Operácia umožňuje prepísať vybranú konfiguráciu inou. V prípade, že cieľová konfigurácia neexistuje, je vytvorená nová.
- **<delete-config>**
  Operácia slúži na zmazanie vybranej konfigurácie (konfiguráciu running nie je možné zmazať).
- **<lock>**
  Operácia umožňuje krátkodobé uzamknutie úložiska konfigurácií.
- **<unlock>**
  Operácia určená na odomknutie uzamknutého úložiska konfigurácií.
- **<kill-session>**
  Operácia umožňuje ukončiť iné ako aktuálne spojenie protokolu NETCONF.
- **<close-session>**
  Operácia určená na ukončenie aktuálneho spojenia.

Obsah

Najvyššia vrstva architektúry komunikačného modelu obsahuje konkrétne konfiguračné dáta, ktorých forma a význam je definovaný výrobcom príslušného zariadenia. Z tohto dôvodu ostáva táto vrstva mimo definície protokolu NECONF.

Rozlišujú sa tri základné typy konfigurácie:

running obsahuje aktuálne konfiguračné dáta, podľa ktorých sú parametre zariadenia momentálne nastavené,

startup obsahuje konfiguračné dáta, podľa ktorých sú parametre zariadenia nastavené pri štarte zariadenia,

candidate slúži ako pracovná kópia konfigurácie running počas jej editácie.

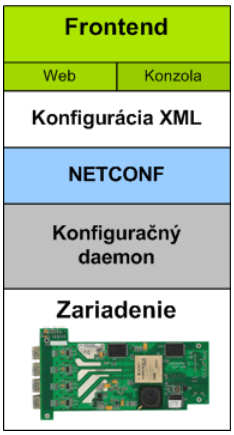
Kapitola 3

Návrh

Kapitola sa zaoberá návrhom konfiguračného systému a návrhom webového konfiguračného rozhrania pre sondu FlowMon.

3.1 Obecný konfiguračný systém

Návrh konfiguračného systému bol inšpirovaný základným princípom dodržiavaným v oblasti návrhu sieťovej infraštruktúry, ktorým je rozdelenie do vrstiev. Podobne ako je tomu v prípade vrstiev modelu komunikačných protokolov TCP/IP, aj vrstvy modelu konfiguračného systému poskytujú služby vyšším vrstvám hierarchie a využívajú pri tom služby vrstiev nižších. Model vrstiev konfiguračného systému je zobrazený na obrázku 3.1.



Obrázok 3.1: Vrstvy konfiguračného systému

Obecne je potrebné počítať s tým, že každé zariadenie má iné požiadavky na konfiguráciu, používa iné nástroje, knižnice, prípadne ovládače. Zapuzdrenie týchto odlišností jednotlivých zariadení má za úlohu **vrstva konfiguračného daemona**. Ten beží ako systémová služba a umožňuje nabootovanie dizajnu, inicializáciu a nastavenie parametrov zariadenia podľa informácií uložených v konfigurácií.

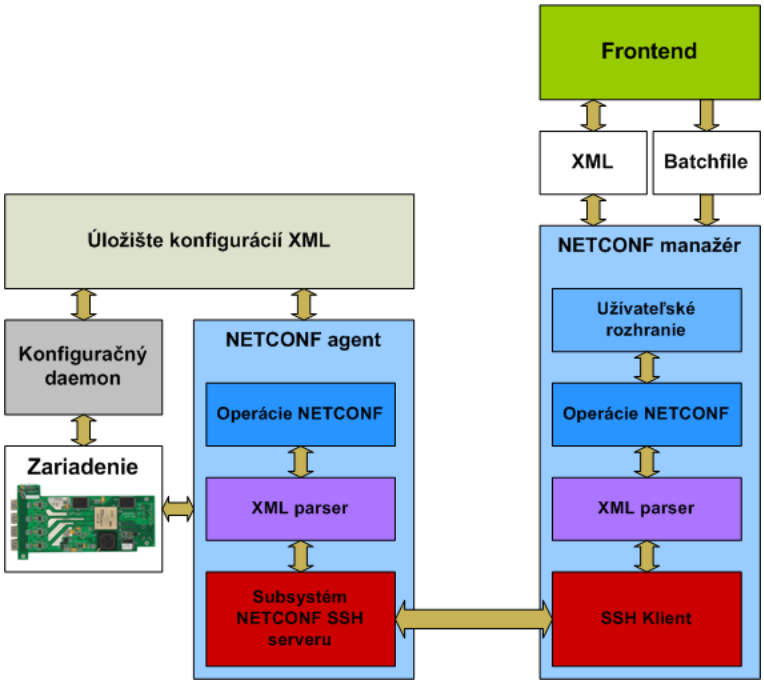
Transportná vrstva je tvorená protokolom NETCONF. Na strane konfigurovaného zariadenia komunikuje NETCONF agent priamo s konfiguračným daemonom. NETCONF manažér komunikuje priamo s konfiguračným frontendom prostredníctvom sady jednoduchých

príkazov.

Dátovú vrstvu tvoria konfiguračné parametre reprezentované vo forme XML. To umožňuje využiť hotové riešenia na ich parsovanie, overovanie podľa schémy a pod. Nástroje na prácu s XML sú dostupné pre väčšinu programovacích jazykov a tak je návrh konfiguračného systému otvorený aj z pohľadu jazyka použitého na implementáciu frontendov.

Editačnú vrstvu tvorí konfiguračný frontend, ktorý slúži na editáciu XML konfigurácie zariadenia. Počíta sa s webovou a konzolovou verziou frontendov.

Schéma konfiguračného systému je znázornená na obrázku 3.2.



Obrázok 3.2: Schéma konfiguračného systému

Z pohľadu implementácie konfiguračných rozhraní pre jednotlivé zariadenia je potrebné pre každé zariadenie vytvoriť:

Schému XML definujúcu štruktúru a obsah konfiguračného XML súboru, pretože nastavované parametre sú špecifické pre každé zariadenie.

Konfiguračného daemona zapuzdrujúcu komunikáciu s príslušným zariadením na úrovni komunikácie s HW.

Frontend editáciu konfiguračných parametrov každého zariadenia je potrebné užívateľovi prezentovať v čo možno najprehľadnejšej forme a preto sa počíta s možnosťou, že každé zariadenie bude mať svoj špecifický frontend.

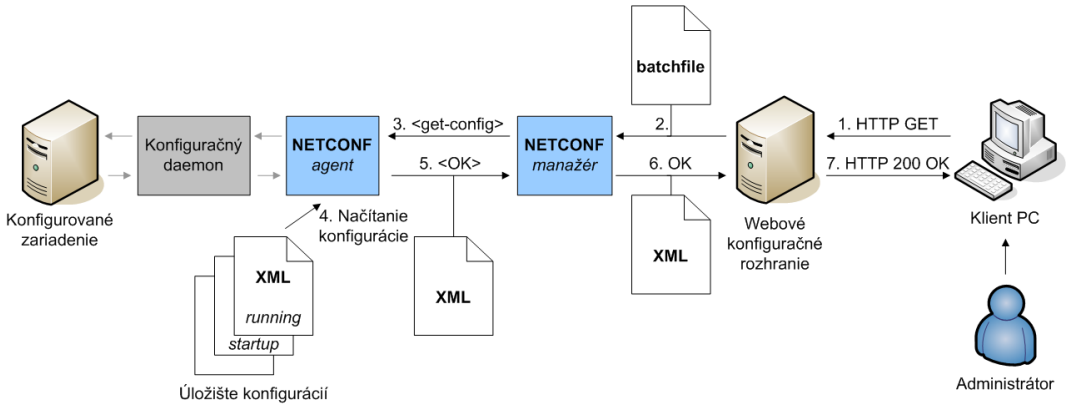
3.2 Využitie protokolu NETCONF

Protokol NETCONF je v rámci konfiguračného systému využitý na správu konfigurácií na strane zariadenia a na transport konfigurácií medzi zariadením a webovým konfiguračným rozhraním.

Webové konfiguračné rozhranie požaduje po protokole nasledovné funkcie:

Načítanie vybranej konfigurácie

Postup pri načítaní vybranej konfigurácie, viď. obrázok 3.3:



Obrázok 3.3: Načítanie konfigurácie pomocou protokolu NETCONF.

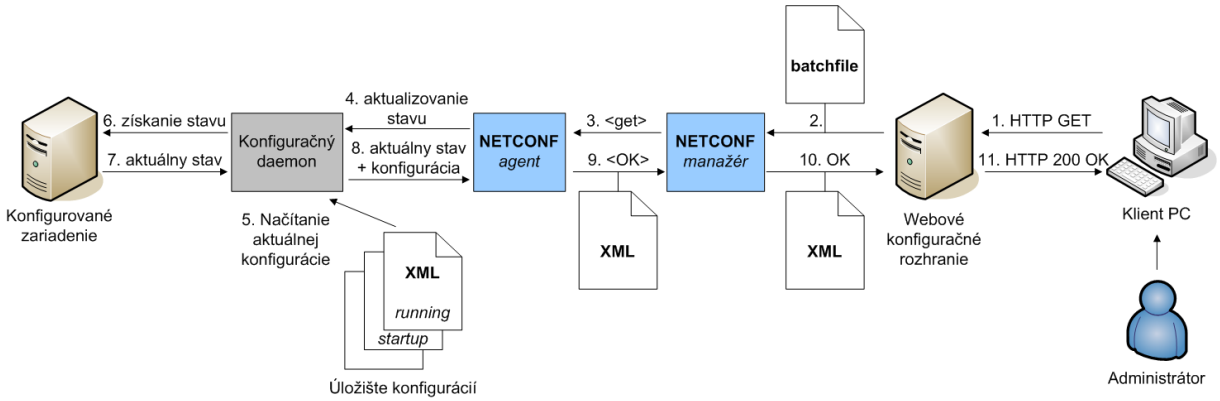
1. Užívateľ odošle požiadavok na stránku webovému rozhraniu.
2. Webové rozhranie vygeneruje NETCONF batchfile s príkazom <*get-config*> s vybranou konfiguráciou. Implicitne načítavaná konfigurácia je *running*. Spustí batchfile pomocou NETCONF manažéra.
3. NETCONF manažér nadviaže SSH spojenie na sieťové zariadenie, spustí NETCONF agenta a vyžiada si vykonanie požadovanej operácie <*get-config*>.
4. NETCONF agent načíta vybranú konfiguráciu.
5. Agent odošle cez otvorené SSH spojenie načítanú konfiguráciu manažérovi. Po úspešnom prenose je SSH spojenie uzatvorené a beh agenta je ukončený [1].
6. NETCONF manažér uloží načítanú konfiguráciu do dočasného súboru s požadovaným názvom a oznámi úspešnosť operácie webovému rozhraniu.
7. Webové rozhranie načíta konfiguráciu z dočasného súboru, vygeneruje stránku, ktorá umožňuje editáciu parametrov konfigurácie a odošle ju klientovi.

Načítanie vybranej konfigurácie a stavových informácií

Postup pri načítaní vybranej konfigurácie a stavových informácií, viď. obrázok 3.4:

1. Užívateľ odošle požiadavok na stránku webovému rozhraniu.
2. Webové rozhranie vygeneruje NETCONF batchfile s príkazom <*get*>. Spustí batchfile pomocou NETCONF manažéra. Operácia get vždy pracuje s konfiguráciou *running*.

[1] SSH spojenie je uzatvorené a NETCONF agent je ukončený po vykonaní posledného príkazu z batchfile, ktorým je v tomto prípade príkaz <*close-session*>



Obrázok 3.4: Načítanie konfigurácie a stavu pomocou protokolu NETCONF.

3. NETCONF manažér nadviaže SSH spojenie na sieťové zariadenie, spustí NETCONF agenta a vyžiada si vykonanie požadovanej operácie <*get*>.

4. NETCONF agent kontaktuje konfiguračného deamona a vyžiada si doplnenie informácií o aktuálnom stave do súboru s aktuálnou konfiguráciou zariadenia.

5. Konfiguračný daemon načíta aktuálnu konfiguráciu *running*.

6. Konfiguračný daemon iniciuje načítanie aktuálneho stavu z registrov zariadenia.

7. Aktuálny stav zariadenia je načítaný.

8. Konfiguračný daemon vygeneruje príslušné XML a doplní ho do aktuálnej konfigurácie. Takto upravenú konfiguráciu odošle NETCONF agentovi.

9. NETCONF agent odošle načítanú konfiguráciu NETCONF manažérovi. Po úspešnom prenose je SSH spojenie uzatvorené a beh agenta je ukončený [2].

10. NETCONF manažér uloží načítanú konfiguráciu do dočasného súboru s požadovaným názvom a oznámi úspešnosť operácie webovému rozhraniu.

11. Webové rozhranie načíta konfiguráciu z dočasného súboru vygeneruje stránku, ktorá umožňuje editáciu parametrov konfigurácie, prípadne zobrazuje stavové informácie zariadenia a odošle ju klientovi.

Uloženie editovanej konfigurácie ako vybranej konfigurácie sondy

Postup pri uložení vybranej konfigurácie, viď. obrázok 3.5:

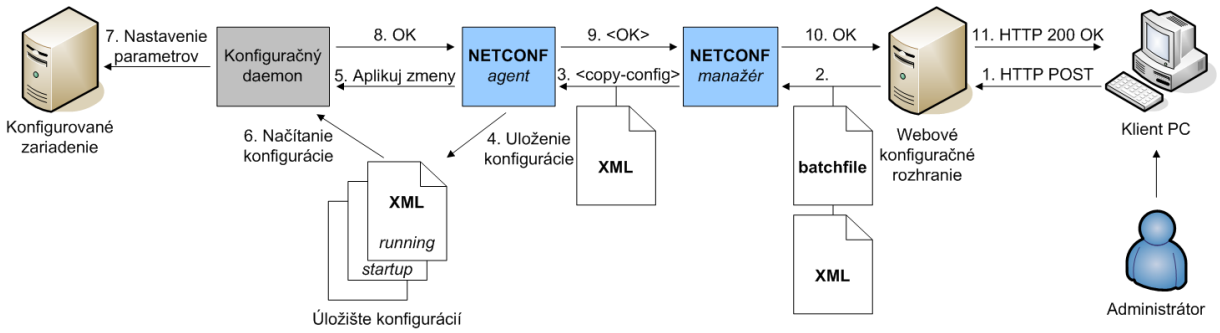
1. Užívateľ odošle vyplnený formulár s nastavením vybraných parametrov zariadenia webovému rozhraniu.

2. Webové rozhranie vygeneruje NETCONF batchfile s príkazom <*copy-config*> s vybranou konfiguráciou. Implicitne je konfigurácia ukladaná ako *running*. Rozhranie

[2] SSH spojenie je uzatvorené a NETCONF agent je ukončený po vykonaní posledného príkazu z batchfile, ktorým je v tomto prípade príkaz <*close-session*>

aktualizuje XML s konfiguráciou a uloží ho do dočasného súboru. Následne spustí batchfile pomocou NETCONF manažéra.

3. NETCONF manažér nadviaže SSH spojenie na sieťové zariadenie, spustí NETCONF agenta a vyžiada si vykonanie požadovanej operácie <*copy-config*>. Zmenenú konfiguráciu mu odošle.
4. NETCONF agent uloží prijatú konfiguráciu ako vybranú konfiguráciu.
5. Agent kontaktuje konfiguračného daemona a vyžiada si nastavenie parametrov zariadenia podľa nového konfiguračného súboru [3].
6. Konfiguračný daemon načíta novú konfiguráciu.
7. Konfiguračný daemon nastaví parametre zariadenia podľa novej konfigurácie.
8. Daemon oznámi NETCONF agentovi úspešnosť aktualizácie nastavenia zariadenia.
9. NETCONF agent oznámi úspešnosť operácie manažérovi. Následne je SSH spojenie uzatvorené a beh agenta je ukončený [4].
10. NETCONF manažér oznámi úspešnosť operácie webovému rozhraniu.
11. Webové rozhranie vygeneruje stránku s oznamom o úspešnosti operácie a odošle ju klientovi.



Obrázok 3.5: Uloženie konfigurácie pomocou protokolu NETCONF.

3.3 Konfiguračný systém pre projekt Liberouter

Špecifikom sieťových zariadení vyvíjaných v rámci projektu Liberouter je, že sa jedná o zariadenia postavené na platforme PC. Použitý operačný systém je Linux, konkrétne distribúcia firmy RedHat. Preto je možné využiť existujúce nástroje a knižnice dostupné pre tento operačný systém, čo by pri používaní proprietárneho hardwaru a operačného systému možné nebolo. To značne uľahčí najmä implementačnú časť vývoja konfiguračného systému na tejto platforme.

[3] Predpokladom je, že zmenenou konfiguráciou bola konfigurácia *running*.

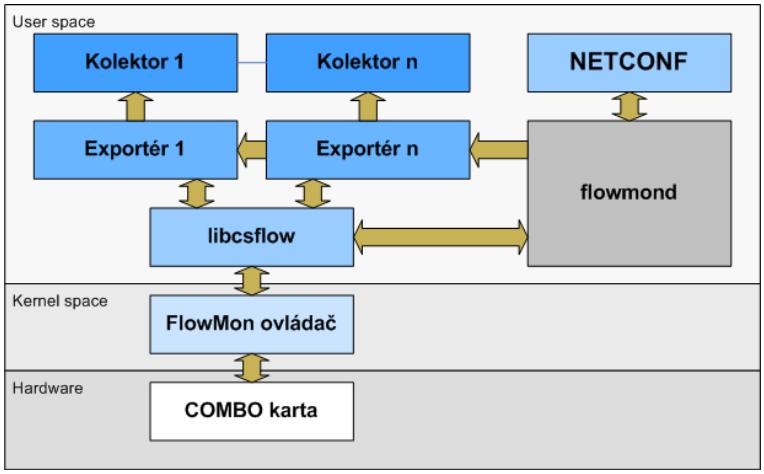
[4] SSH spojenie je uzatvorené a NETCONF agent je ukončený po vykonaní posledného príkazu z batchfile, ktorým je v tomto prípade príkaz <*close-session*>

Editačná vrstva je v prípade webového rozhrania tvorená aplikáciou v PHP, ktorá využíva služby servera apache. Funkcie transportnej vrstvy zabezpečuje vlastná implementácia protokolu NETCONF vyvinutá v rámci projektu Liberouter. Ako transportný protokol používa protokol SSH. Konfiguračný daemon je vyvíjaný pre každé zariadenie zvlášť. Slúži na inicializáciu firmware a komunikáciu s COMBO kartou. Parsuje XML konfiguráciu a nastavuje príslušné parametre zariadenia. Ukážkový príklad konfigurácie sondy FlowMon je uvedený v dodatku B.

3.3.1 Konfigurácia sondy FlowMon

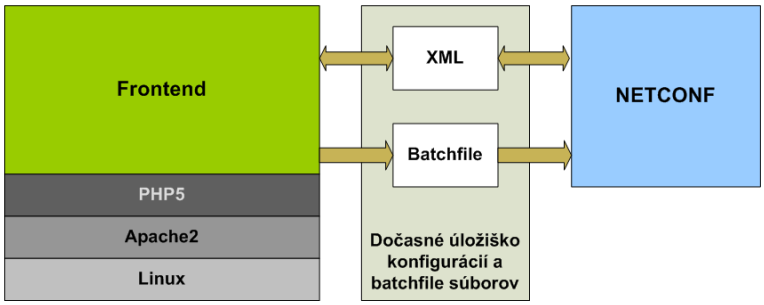
Konfiguračný systém navrhnutý pre sondu FlowMon je postavený na základe definovanom pri návrhu obecného konfiguračného systému. Súčasťou tejto bakalárskej práce je implementácia časti webového forntendu. Implementácia protokolu NETCONF je predmetom bakalárskej práce Radka Krejčího, konfiguračný daemon *flowmond* bol naprogramovaný Petrom Špringlom.

Detail časti konfiguračného systému na strane zariadenia sondy FlowMon je na obrázku 3.6.



Obrázok 3.6: Konfiguračný systém na strane sondy

Schéma časti konfiguračného systému na strane webového frontendu je na obrázku 3.7.

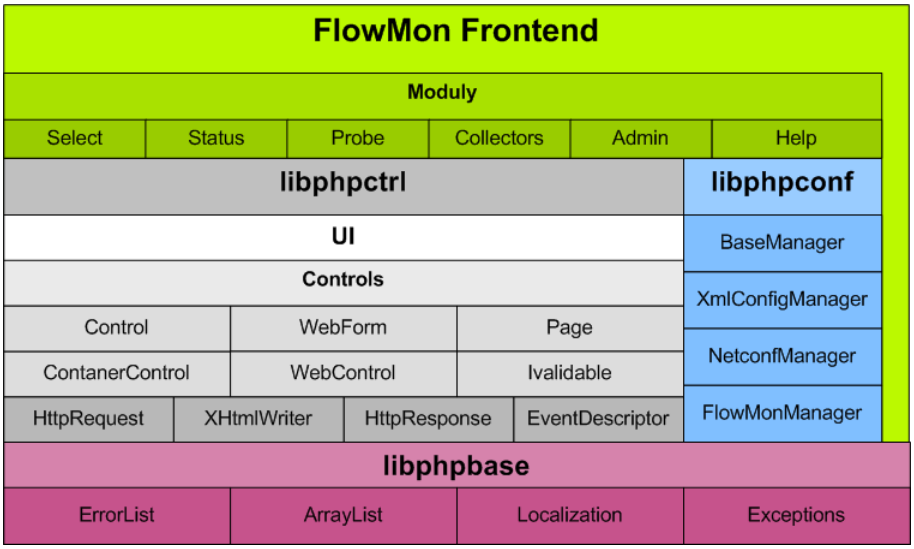


Obrázok 3.7: Konfiguračný systém na strane webového frontendu

3.4 FlowMon Frontend

Návrh webového konfiguračného rozhrania pre sondu FlowMon počíta s tým, že v budúcnosti sa budú vytvárať webové konfiguračné rozhrania pre ďalšie sieťové zariadenia vyvíjané v rámci projektu Liberouter. Preto bola zvolená modulárna koncepcia a boli vytvorené obecné knižnice na prácu s GUI v PHP a knižnica zapuzdrujúca komunikáciu so sondou FlowMon pomocou protokolu NETCONF. Aplikáciu *FlowMon Web Configuration Interface* je možné považovať za vzorovú implementáciu webového frontendu založeného na obecnom konfiguračnom systéme navrhnutom v rámci tejto bakalárskej práce.

Celková schéma webového frontendu vrátane všetkých súčastí implementovaných v knižniciach *libphpbase*, *libphpconf* a *libphpctrl* je zobrazená na obrázku 3.8.



Obrázok 3.8: Celková schéma súčastí webového frontendu pre sondu FlowMon

Kapitola 4

Implementácia

Kapitola popisuje implementačné detaily webového konfiguračného rozhrania pre sondu FlowMon. Boli vytvorené tri knižnice a jedna webová aplikácia.

4.1 libphpctrl

Kvôli zjednodušeniu práce s GUI v PHP bola navrhnutá a implementovaná knižnica *libphpctrl*. Ako inšpirácia pri návrhu poslúžili webové technológie ASP.NET spoločnosti Microsoft. Knižnica rieši základné problémy práce s GUI v PHP:

- Základná sada prvkov GUI, generovanie HTML kódu,
- Udržovanie stavu medzi jednotlivými požiadavkami,
- Udalosťami riadené programovanie,
- Validácia vstupov, chybové výpisy.

4.1.1 Prvky GUI

Jednotlivé užívateľské prvky možno rozdeliť do dvoch skupín:

- Jednoduché užívateľské prvky,
- Zložené užívateľské prvky - kontajnery.

Jednoduché užívateľské prvky ako vstupné textové pole *TextBox*, popisok *Label* atď. môžu byť súčasťou kontajnerových užívateľských prvkov ako *Panel*, webový formulár *WebForm*. Stránka je tvorená stromovou hierarchiou týchto prvkov. Koreňový uzol stromu je vždy tvorený webovým formulárom, listy sú tvorené jednoduchými užívateľskými prvkami. Knižnica bola vytváraná najmä na uľahčenie a sprehľadnenie práce s formulármi pri vytváraní webového konfiguračného rozhrania pre sondu FlowMon a preto neimplementuje všetky užívateľské prvky, ktoré jazyk HTML definuje.

Control: Z abstraktnej triedy *Control* sú odvodené triedy všetkých ostatných užívateľských prvkov. Samotná trieda *Control* implementuje prácu s udalosťami, prácu s hierarchickým stromom užívateľských prvkov na stránke, udržuje si referencie na nadradený a podradené užívateľské prvky a referencie na formulár *WebForm* a stránku *Page*, do ktorej je prvok začlenený. Trieda *Control* taktiež rieši jednoznačnú identifikáciu

užívateľského prvku na stránke a má deklarované abstraktné metódy na inicializáciu stavu prvku *Initialize* a spracovanie vstupu od užívateľa *ProcessPostBack* a generovanie HTML kódu *Render*.

WebControl: Abstraktná trieda *WebControl*, ktorá je odvodená od triedy *Control* je bázovou triedou všetkých tried jednoduchých užívateľských prvkov. Riadi inicializáciu ich stavu, načítanie vstupu a generovanie HTML kódu.

ContainerControl: Trieda *ContainerControl* naopak riadi inicializáciu stavu, načítanie vstupu a generovanie HTML kódu zloženého užívateľského prvku a všetkých vnorených prvkov, ktoré obsahuje. Dôležitým atribútom zložených prvkov je *OwnRender*, ktorý ovplyvňuje samotnú generáciu HTML. Ak je nastavená hodnota *false* (východzia hodnota), generuje sa nielen HTML kód zloženého prvku ale aj kód všetkých jemu podriadených prvkov. V opačnom prípade sa generuje iba kód zloženého prvku a na vygenerovanie kódu jemu priamo podriadených prvkov je potrebné volať ich metódu *Render* explicitne. To umožňuje zasadiť tieto prvky do kontextu stránky na ľubovoľné miesto (viď. kapitolu 4.1.2)

4.1.2 Systém generovanie HTML kódu

Na zjednodušenie generovania HTML kódu, ktorý spĺňa nároky špecifikácie *XHTML 1.0 Strict* bola vytvorená trieda *XHtmlWriter*. Spravuje výstupný buffer, do ktorého je zapisovaný kód, ktorý pomocou nej generujú jednotlivé užívateľské prvky v metóde *Render*. Samotné generovanie kódu v metóde *Render* prebieha nasledovne:

1. Získanie referencie na inštanciu *XHtmlWriter*-a priradeného aktuálnej odpovedi. Získa sa zavolaním metódy *GetWriter* triedy *HttpResponse*. Referenciu na aktuálnu odpoveď je možné získať volaním metódy *GetResponse* triedy *Page*.

2. Zavolanie chránenej metódy *RenderControl*, ktorá v prípade, že je prvok viditeľný (atribút *Visible* = *true*) postupne zavolá metódy:

   RenderStart: inicializuje kolekciu atribútov elementu HTML, zapíše počiatočnú HTML značku a atribúty elementu.

   RenderContent: zapíše obsah elementu HTML.

   RenderEnd: uzatvorí element zápisom koncovej značky.

3. Zapísanie obsahu buffera *XHtmlWriter*-a do buffera odpovede zasielanej klientovi.

Implementácia generovania HTML kódu v *libphpctrl* umožňuje obísť potrebu šablónovacieho systému a súčasne zachováva možnosť vkladať užívateľské prvky na ľubovoľné miesto v kontexte stránky. Možnosť pridania šablónovacieho systému do *libphpctrl* však nie je do budúcnosti vylúčená. Aby bolo možné aj v súčasnosti oddeliť aplikačnú logiku od prezentačnej, je každá stránka tvorená dvoma súbormi:

- ***názov-stránky*.php** - súbor obsahujúci PHP kód s prezentačnou logiku,
- ***názov-stránky*_code.php** - súbor obsahujúci PHP kód s aplikačnou logiku.

4.1.3 Systém obsluhy požiadavkov

Aplikácie v PHP sú založené na princípe dotaz - odpoveď, anglicky request - response. Knižnica *libphpctrl* definuje triedu *Page*, ktorá s pomocou tried *HttpRequest* a *HttpResponse* zapuzdruje komunikáciu s klientom. Umožňuje prácu s dátami odoslanými pomocou metódy HTTP POST a HTTP GET protokolu HTTP a spravuje výstupný buffer, do ktorého sa ukladá HTML kód generovaný jednotlivými užívateľskými prvkami formulárov.

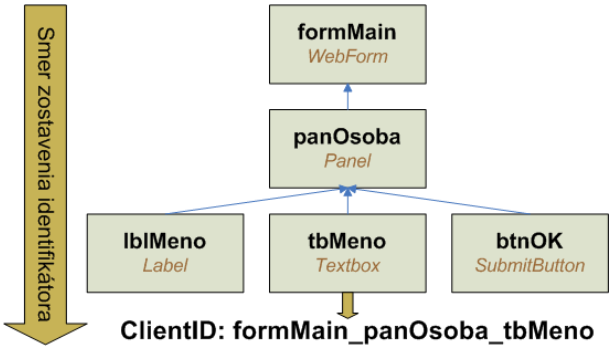
Pre každý požiadavok od klienta sa vytváraná nová inštancia triedy *Page*. Knižnica *libphpctrl* sa stará o inicializáciu jednotlivých prvkov GUI, sprístupňuje postback dáta a zúčastňuje sa na tvorbe HTML kódu odpovede. Diagram znázorňujúci obsluhu jedného dotazu klienta je znázornený na obrázku v dodatku C.1.

4.1.4 Udržovanie stavu

Základné grafické užívateľské prvky formulárov libphpctrl si dokážu uchovávať svoj stav medzi jednotlivými požiadavkami. Využívajú na to dva základné atribúty formulárových prvkov input jazyka HTML:

- *ID, NAME* - na uloženie identifikátora prvku,
- *VALUE* - na uloženie aktuálnej editovanej hodnoty prvku.

Atribút obecného užívateľského prvku *ID* definovaný v bázovej triede *Control* slúži na jednoznačnú identifikáciu prvku v rámci jednej úrovne hierarchického stromu prvkov stránky. Zároveň je využívaný metódou *ComposeClientID* triedy *Control* na vytvorenie jednoznačného identifikátora v rámci celej stránky *ClientID*. Tento identifikátor je použitý ako hodnota atribútu *NAME* input prvkov jazyka HTML. Pomocou metódy *FindControlByID* triedy *Control* je možné získať referenciu na prvok v rámci jednej úrovne hierarchického stromu prvkov, na získanie referencie v rámci celého stromu slúži metóda *FindControlByClientID*. Proces zostavenia *ClientID* je znázornený na obrázku 4.1.



Obrázok 4.1: Zostavenie identifikátoru *ClientID*

4.1.5 Udalosti

Udalosť je akýkoľvek presne rozlíšiteľný jav, ktorý môže nastať. Z pohľadu programátora je dôležité, aby mu jeho programovacie prostriedky umožňovali na vzniknutú udalosť nejakým ním definovaným spôsobom zareagovať. Táto reakcia je umožnená vytvorením špeciálnej metódy zvanej *event handler*, ktorá je vyvolaná pri vzniku udalosti.

Knižnica *libphpctrl* umožňuje programátorovi vytvárať ľubovoľné udalosti a zaisťuje ich vyvolanie pomocou metód na prácu s udalosťami v bázovej triede *Control*. Postup pri vytváraní novej udalosti a jej následnom použití v programe:

1. Vytvorenie metódy *<názov_udalosti>($sender, $param)*, ktorá zaistí vyvolanie všetkých zaregistrovaných *event handlerov*. Hodnota parametra *$sender* je nastavená na inštanciu objektu, ktorý udalosť vyvolal (obyčajne *this*). Parameter *$param* slúži na prenos rozširujúcich informácií, v prípade, že to obsluha udalosti vyžaduje. Táto metóda je volaná v momente výskytu udalosti v systéme.

2. Vytvorenie metódy *event handlera*, ktorá obsahuje zdrojový kód, ktorý sa má vykonať v čase výskytu udalosti.

3. Registrovanie novo vytvorenej metódy *event handlera* pomocou volania metódy *AttachEventHandler($name, $handler)* triedy *Control*, ktorá uloží ukazovateľ na túto metódu (parameter *$handler*) do kolekcie *event handlerov* definovanej pre danú udalosť identifikovanú jej názvom (parameter *$name*). Metódy uložené v tejto kolekcií sú postupne volané v čase keď udalosť nastane. Na ich vyvolanie slúži metóda *FireEvent* triedy *Control*. Odregistrovať registrovaný *event handler* je možné volaním metódy *DetachEventHandler* triedy *Control*. Registrácia obvykle prebieha v tele konštruktora objektu.

Knižnica *libphpctrl* umožňuje využiť dva spôsoby vyvolania udalosti v systéme a následného spustenia *event handlerov* registrovaných pre danú udalosť. Použitý variant záleží na implementácií metódy *<názov_udalosti>($sender, $param)* a teda je plne v rukách programátora, ktorý danú udalosť vytvára. Možné varianty volania sú:

- Okamžité vyvolanie registrovaných *event handlerov*,
- Odložené (*delayed*) vyvolanie registrovaných *event handlerov*.

Na okamžité vyvolanie slúži metóda *FireEvent($name, $sender, $param)* triedy *Control*, ktorá zaistí okamžité volanie všetkých registrovaných metód z kolekcie *event handlerov* udalosti určenej identifikátorom *$name*.

Odložené vyvolanie je umožnené volaním metódy *AddFiredEvent($descriptor)* triedy *WebForm*, ktorá zaznamená všetky výskyty danej udalosti. *Event handlery* priradené danej udalosti sú vyvolané až po spracovaní *postback-u*.

4.1.6 Validácia vstupov

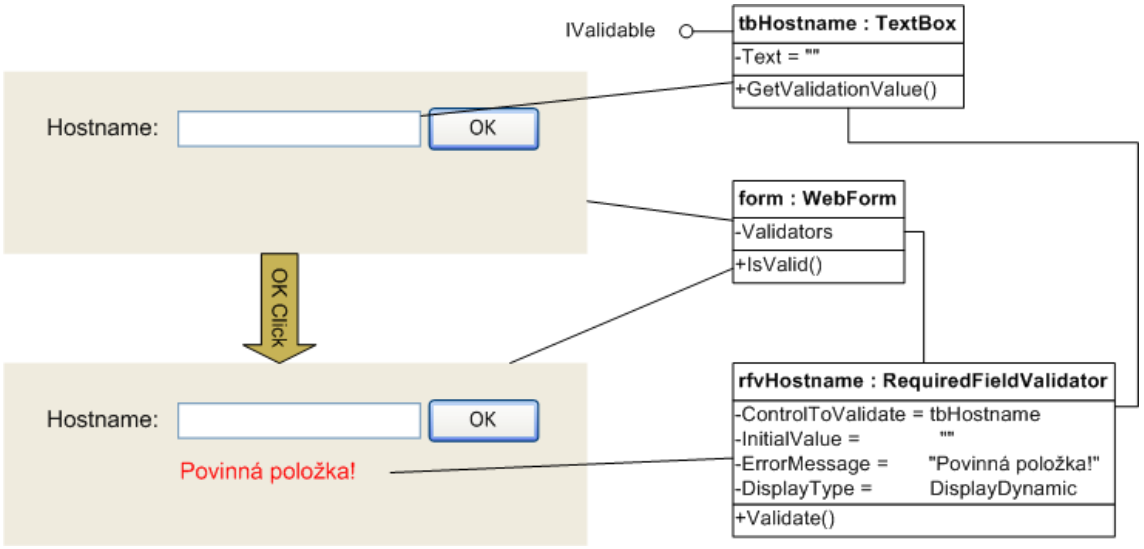
Vstupné dáta ktoré zadáva užívateľ je potrebné kvôli bezpečnosti dôsledne kontrolovať. Prvky grafického užívateľského rozhrania knižnice *libphpctrl*, ktoré umožňujú zadávanie vstupu od užívateľa preto implementujú rozhranie *IValidable*. Metóda *GetValidationValue* rozhrania *IValidable* slúži na získanie aktuálnej užívateľom zadanej hodnoty. Jej správnosť je možné kontrolovať pomocou validátorov. Validátor je špeciálny prvok formulára, ktorý dokáže v prípade chybného vstupu automaticky zobraziť chybovú správu po odoslaní formuláru na spracovanie serveru. Knižnica *libphpctrl* má implementované dva základné typy validátorov:

RequiredFieldValidator kontroluje vyplnenie povinných formulárových polí,

DataTypeValidator kontroluje, či vyplnené dáta zodpovedajú požadovanému typu (číslo, IP adresa...).

V prípade potreby je možné definovať si vlastné validátori odvodením od triedy *BaseValidator* a predefinovaním jej metódy *ValidateValue*.

Kontrola správnosti všetkých údajov zadávaných užívateľom je vyvolaná volaním metódy *IsValid* triedy *WebForm*. Ak boli všetky vstupné dáta zadané správne, vracia metóda hodnotu *true*, v opačnom prípade *false*. Ošetrenie oboch stavov je plne v rukách programátora. Proces validácie je znázornený na obrázku 4.2 (položka *Hostname* je povinná a nebola vyplnená).



Obrázok 4.2: Proces validácie položiek formulára

4.2 libphpconf

Konfiguračný systém, tak ako bol navrhnutý, kladie na konfiguračné frontendy niekoľko požiadaviek:

- načítanie, editácia a ukladanie konfigurácie v XML,
- komunikácia s konfiguračným daemonom prostredníctvom protokolu NETCONF.

Webové konfiguračné rozhrania musia navyše podporovať:

- editáciu viacerými užívateľmi súčasne a s tým súvisiacu
- správu pomocných konfiguračných XML súborov (pre jednotlivé *session*).

Návrh konfiguračného systému, ktorý do budúcnosti počíta s vytváraním nových webových konfiguračných rozhraní pre viaceré sieťové zariadenia, viedol k vytvoreniu knižnice *libphpconf*, ktorá rieši správu konfiguračných XML súborov, komunikáciu pomocou protokolu NETCONF a navrhuje koncepciu tzv. *manažérov*. Každý manažér rieši svoju konkrétnu

úlohu, pre ktorú bol vytvorený (napr. *NetconfManager* komunikáciu pomocou protokolu NETCONF) a svoje služby poskytuje ostatným manažérom, ktorý jeho služby využívajú.

V takto vzniknutej hierarchickej stromovej štruktúre vzájomne prepojených manažérov, je potrebné navrhnúť efektívnu správu chybových stavov. Preto bázová trieda *BaseManager* definuje kolekciu na uloženie chybových a informačných správ. Tieto správy sa postupne šíria v hierarchií smerom nahor, čo dáva užívateľskému rozhraniu možnosť pracovať iba s kolekciou správ koreňového manažéra a tie následne zobrazovať užívateľovi. Knižnica *libphpctrl* má na zobrazovanie správ od manažérov vytvorený špeciálny grafický užívateľský prvok *ListErrorsControl*.

UML diagram tried knižnice *libphpconf* je zobrazený na obrázku v dodatku D.2.

4.3 libphpbase

Knižnica *libphpbase* obsahuje triedy využívané súčasne knižnicami *libphpctrl* a *libphpconf*. UML diagram tried knižnice *libphpbase* je zobrazený na obrázku v dodatku D.1.

4.4 FlowMon Web Configuration Interface

Webové konfiguračné rozhranie sondy FlowMon bolo vytvorené s použitím knižnice *libphpctrl* a *libphpconf*. Slúži na nastavenie základných parametrov sondy ako napríklad timeouty, sampling, logovanie, adresy porty a protokoly kolektorov a podobne.

Koncepcia konfiguračného rozhrania bola navrhnutá ako jednotná obálka, do ktorej sa načítavajú jednotlivé *moduly* rozhrania. Každý modul slúži na editáciu špecifickej časti konfigurácie, zobrazuje informácie, alebo umožňuje vykonávať administratívne úlohy. Rozhranie sa môže nachádzať v niekoľkých možných stavoch:

Select zatiaľ nebolo vybrané zariadenie, ktoré sa bude konfigurovať,

Disabled zariadenie bolo vybrané, ale nepodarilo sa načítať jeho konfiguráciu,

Enabled zariadenie bolo vybrané, konfigurácia je načítaná a jej editácia je povolená.

Stav rozhrania ovplyvňuje prístupnosť jednotlivých modulov na používanie užívateľom (viď obrázok 4.3). Moduly vyznačené zelenou farbou sú prístupné, moduly vyznačené červenou farbou sú v danom stave neprístupné.

Frontend		
Select	**Disabled**	**Enabled**
Select	Select	Select
Status	Status	Status
Probe	Probe	Probe
Collectors	Collectors	Collectors
Admin	Admin	Admin
Help	Help	Help

Obrázok 4.3: Prístupnosť modulov v závislosti na stave frontendu

4.4.1 Moduly konfiguračného rozhrania

Select: Modul slúži na výber konfigurovaného zariadenia. Užívateľ zadáva *hostname* alebo *IP adresu* zariadenia, *užívateľské meno* a *heslo* pod ktorým sa má konfiguračné rozhranie na sondu pripojiť. Po vyplnení a odoslaní dát je dotaz automaticky presmerovaný do modulu *Status*. Modul je do budúcnosti možné vylepšiť o možnosť uchovávať si históriu konfigurovaných zariadení, čo by učinilo aplikáciu užívateľsky prívetivejšou.

Status: Modul je určený na prehľadné zobrazenie stavu zariadenia a samotného frontendu. Aktuálne zobrazuje informácie o:

- editovanej konfigurácií,
- prostredí,
- protokole NETCONF.

Do budúcnosti sa počíta so zobrazovaním stavu sieťových rozhraní karty, prípadne aj grafickými výstupmi z kolektorov.

Probe: Modul slúži na editáciu základných parametrov sondy. Je rozdelený na tri sekcie:

Timeouts nastavenie aktívneho a neaktívneho časovača,

Sampling nastavenie samplingu[1],

Logging nastavenie logovania informácií.

V budúcnosti nebude problém modul rozšíriť o podporu nastavovania nových konfiguračných parametrov sondy.

Collectors: Umožňuje nastaviť parametre kolektorov, ako *hostname* alebo *IP adresa*, *port*, použitý *protokol* a čas po ktorom sa majú odosielať údaje na kolektor *template send timeout*. Podľa nastavenia kolektorov spúšťa konfiguračný daemon príslušné exportéri.

Do budúcnosti sa počíta s podporou možnosti nastavovania filtrov pre jednotlivé kolektory.

Admin: Modul umožňuje administrátorovi načítať vybranú konfiguráciu sondy, prípadne nahrať aktuálne editovanú konfiguráciu ako vybranú konfiguráciu sondy. Pre účely zálohy nastavení je možné vyexportovať aktuálne editované nastavenie ako XML súbor, ktorý je možné uložiť pomocou webového prehliadača. Funkcia import umožňuje načítanie konfigurácie z XML súboru odoslaného pomocou webového prehliadača a jej následnú editáciu.

Help: Obsahuje odkazy na webové stránky projektu Liberouter, stránky zamerané na podporu sondy FlowMon a HTML verziu README súboru dodávaného v balíčku spoločne s programovým vybavením sondy.

[1] Vynechávanie paketov v prípade, že sonda nestíha spracovávať daný dátový tok.

4.4.2 Editácia konfigurácie

Aplikácia umožňuje konfiguráciu viacerých zariadení z jedného miesta viacerými užívateľmi súčasne. Typický scenár použitia je nasledovný:

1. Načítanie *running* konfigurácie sondy s využitím protokolu NETCONF.
2. Uloženie načítanej konfigurácie do dočasného súboru na server.
3. Editácia dočasne uloženej konfigurácie.
4. Uloženie zmenenej konfigurácie na sondu pomocou protokolu NETCONF.

Kvôli vzájomnému rozlíšeniu jednotlivých požiadavkov sú pre každú *session* v pracovnom adresári aplikácie vytvorené dva súbory s jedinečným názvom:

<unique-id>.xml súbor s konfiguráciou,

<unique-id>.netconfbatch batchfile s príkazmi pre NETCONF manažér.

Dočasné súbory sú z pracovného adresára odstraňované automaticky, po uplynutí určenej doby od ich poslednej zmeny.

Kapitola 5

Záver

V rámci práce bola naštudovaná problematika hardwarovej a softwarovej architektúry zariadení vytvorených v rámci projektu Liberouter. Podrobne bola naštudovaná najmä problematika hardwarovej a softwarovej architektúry pasívnej monitorovacej sondy FlowMon, pre ktorú bolo následne navrhnuté a vytvorené webové konfiguračné rozhranie.

V rámci analýzy a návrhu obecného konfiguračného systému pre sieťové zariadenia bola naštudovaná problematika protokolu NETCONF, jeho architektúra ako aj práca s konkrétnou implementáciou protokolu NETCONF vyvinutou v rámci projektu Liberouter. Následne bol navrhnutý obecne použiteľný konfiguračný systém, založený na protokole NETCONF. Systém počíta s možnosťou konfigurácie cez webové rozhranie. Obecný návrh konfiguračného systému bol následne konkretizovaný do podoby priamo využiteľnej so zariadeniami vyvíjanými v rámci projektu Liberouter.

Detailne boli naštudované a analyzované najmä potreby a možnosti konfigurácie sondy FlowMon. Následne bol navrhnutý konfiguračný systém pre sondu FlowMon, ktorý je postavený na základoch obecného konfiguračného systému. v rámci systému pre sondu FlowMon bolo navrhnuté webové konfiguračné rozhranie, ktoré umožňuje editovať konfiguráciu sondy.

Navrhnuté riešenie bolo implementované do podoby funkčnej webovej aplikácie. Na implementáciu bol zvolený jazyk PHP. V rámci riešenia boli vytvorené znovu použiteľné knižnice tried *libphpbase*, emphlibphpconf a *libphpctrl*. Knižnica *libphpconf* obsahuje triedy na prácu s protokolom NETCONF, s konfiguráciou v XML a umožňuje ovládať načítanie a ukladanie konfigurácie sondy FlowMon. Knižnica *libphpctrl* podporuje objektovo orientovaný návrh grafického užívateľského rozhrania v jazyku PHP. Do budúcnosti sa uvažuje o rozšírení aplikácie *FlowMon Web Configuration Interface* o podporu konfigurácie parametrov softwarovej sondy *fprobe*, ktorá podobne ako hardwarové riešenie FlowMon slúži na zber NetFlow dát na sieti.

Funkcionalita webového konfiguračného rozhrania a jeho spolupráca s protokolom NETCONF a konfiguračným daemonom bola úspešne otestovaná na serveroch projektu Liberouter. V súčasnosti sa pripravuje vydanie nového balíčku s programovým vybavením pre sondu FlowMon, ktorého súčasťou bude aj webové rozhranie navrhnuté a implementované v rámci tejto bakalárskej práce.

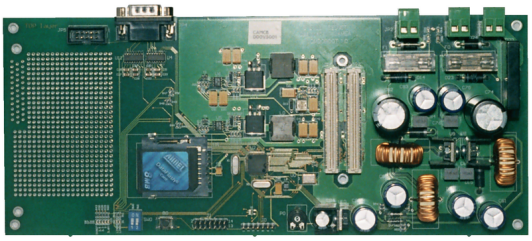
Dodatok A

Hardwarová výbava kariet COMBO

A.1 Hlavné karty

COMBO-BOOT

- 2xPower supply (input voltage 12-30V)
- TI processor - MSP430FI49IPM
- RS232 interface
- Connector for add-on cards



COMBO-PTM

- Spartan3 - X3S200 (alternatively X3S400)
- TI processor - MSP430FI49IPM
- PCI interface chip PCI9054
- CLPD - XCR3256XL
- 1xEEPROM - 93S66
- 2xRS232/485 interface
- Extension/test connector

COMBO6

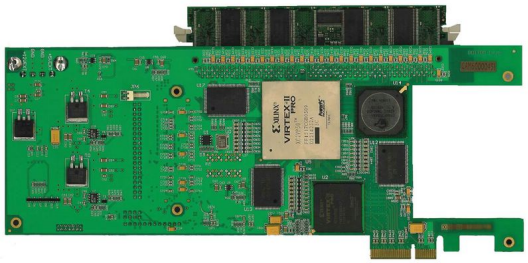
- Virtex II - XC2V3000 (the series up to XC2V6000 can be used)
- PCI interface chip PCI9054
- CLPD - XCR3256XL
- CAM - 2Mb ternary CYNSE70064A
- 3xSSRAM - 2MB 512K36
- EEPROM - 93S66
- PCI connector
- Connector for add-on card
- Extension/test connector
- DRAM connector for PC DDR up 2GB

COMBO6X

- Virtex II PRO - XC2VP50 (the series up to XC2V70 can be used)
- Virtex II PRO - XC2VP4 with PCI core (alternatively PCI-X)
- CAM - 2Mb ternary CYNSE70064A
- 3x SSRAM - 2MB 512K36
- EEPROM - 93S66
- PCI connector
- Connector for add-on card
- Extension/test connector
- DRAM connector for PC DDR up 2GB

COMBO6E

- Virtex II PRO - XC2VP50 (the series up to XC2V70 can be used)
- Virtex II PRO - XC2VP20 with Express PCI core
- CAM - 2Mb ternary CYNSE70064A
- 3x SSRAM - 2MB 512K36
- EEPROM - 93S66
- Express PCI connector
- Connector for add-on card
- Extension/test connector
- DRAM connector for PC DDR up 2GB



A.2 Rozširujúce karty

COMBO4-MTX

- Virtex II - XC2V1000 (the series up to XC2V3000 can be used)
- 2xSSRAM - 2MB 512K36
- 3xEEPROM - 93S66
- 4xPhyter - DP83865
- 4xRJ45 connectors
- Connector to the mother card

COMBO4-SFP

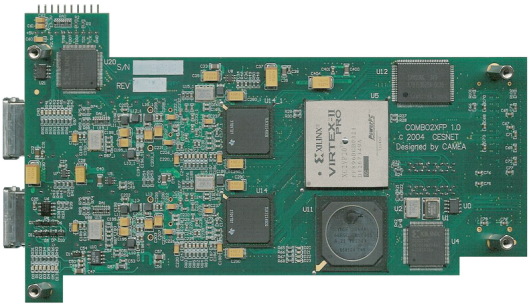
- Virtex II - XC2V1000 (the series up to XC2V3000 can be used)
- 2xSSRAM - 2MB 512K36
- 3xEEPROM - 93S66
- 4xSerdes - VSC7145
- 4xSFP cages
- Connector to the mother card

COMBO4-SFPRO

- Virtex II - XC2VP20 (the series up to XC2VP30 can be used)
- 2xSSRAM - 2MB 512K36
- 1xEEPROM - 93S66
- 4xXFP cages
- Connector to the mother card

COMBO2-XFP

- Virtex II - XC2VP20 (the series up to XC2VP30 can be used)
- SSRAM - 2MB 512K36
- 3xEEPROM - 93S66
- 2xSet of deser chips
- 2xXFP cages
- Connector to the mother card



Dodatok B

Príklad XML konfigurácie sondy FlowMon

```
<flowmon-config xmlns=‘‘http://www.liberouter.org/ns/netopeer/flowmon/1.0’’>
    <probe>
        <active-timeout>120000</active-timeout>
        <inactive-timeout>5000</inactive-timeout>
    <sampling>
        <sampling-rate>10</sampling-rate>
        <sample-and-hold-rate>100</sample-and-hold-rate>
    </sampling>
    <logging>
        <log-destination>
            <syslog-host content-type=‘‘ipv4’’>10.1.2.3</syslog-host>
        </log-destination>
        <log-level>info</log-level>
    </logging>
    </probe>
    <collectors>
        <collector>
            <description>First collector</description>
            <host content-type=‘‘ipv4’’>192.168.2.3</host>
            <udp-port>6378</udp-port>
            <protocol>netflow-v5</protocol>
        </collector>
        <collector>
            <protocol>netflow-v9</protocol>
            <description>Second collector (IPv6 enabled)</description>
            <host content-type=‘‘ipv6’’>2001:1234:5678:9ABC::D</host>
            <udp-port>60000</udp-port>
            <template-send-period>30</template-send-period>
        </collector>
    </collectors>
</flowmon-config>
```

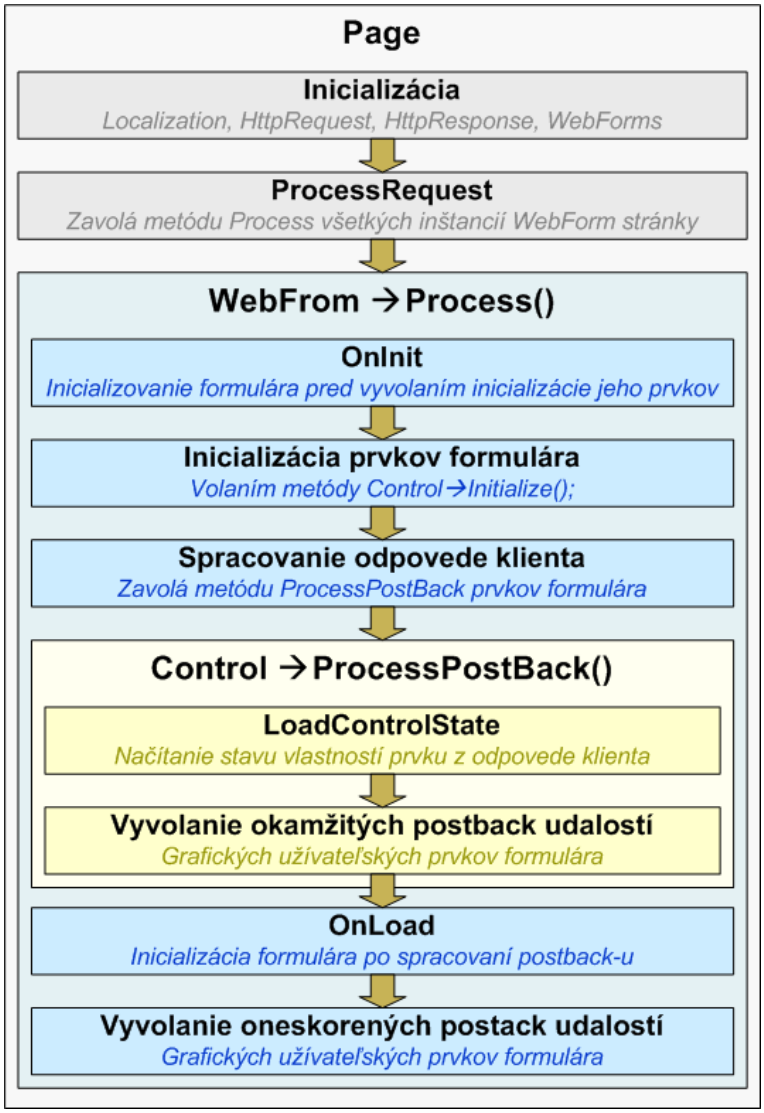
Dodatok C

Spracovanie požiadavku klienta

C.1 Spracovanie požiadavku klienta

Obrázok C.1: Spracovanie požiadavku klienta

C.2 Detail spracovania požiadavku knižnicou *libphpctrl*

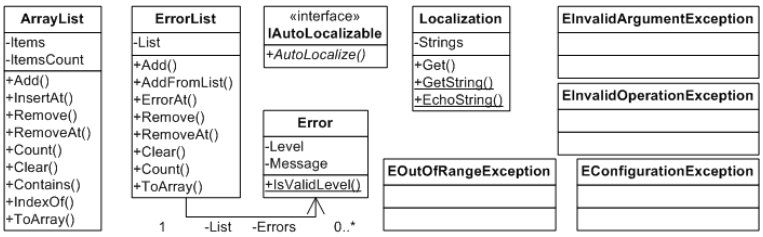


Obrázok C.2: Detail spracovania požiadavku knižnicou *libphpctrl*

Dodatok D

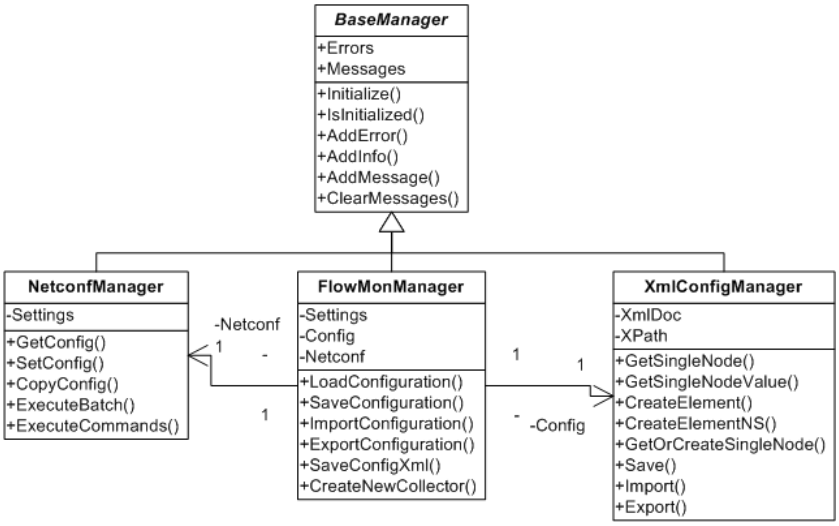
UML Diagramy

D.1 Diagram tried knižnice *libphpbase*



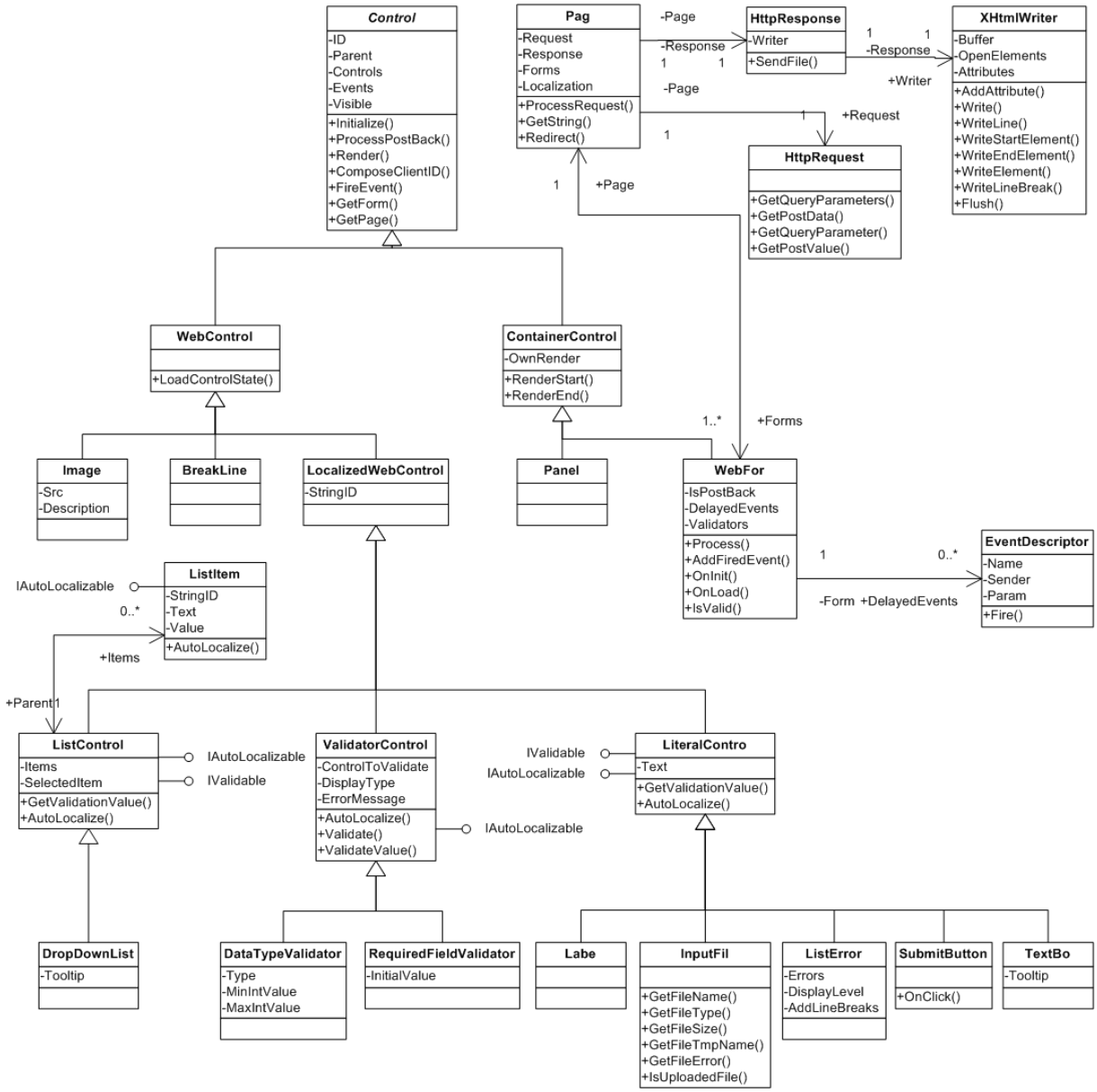
Obrázok D.1: Diagram tried knižnice *libphpbase*

D.2 Diagram tried knižnice *libphpconf*



Obrázok D.2: Diagram tried knižnice *libphpconf*

D.3 Diagram tried knižnice *libphpctrl*



Obrázok D.3: Diagram tried knižnice *libphpctrl*

Literatúra

[1] Radek Krejčí;
Konfigurace síťových zařízení protokolem NETCONF.
Masarykova Univerzita, Fakulta Informatiky,
2006.

[2] Pavel Čeleda, Milan Kováčik, Tomáš Koníř, Vojtěch Krníček, Petr Špringl, Martin Žádnik;
FlowMon Probe.
http://www.cesnet.cz/doc/techzpravy/2006/flowmon-probe/.

[3] Liberouter project.
http://www.liberouter.org/.

[4] Liberouter.
http://www.liberouter.org/liberouter.php.

[5] SCAMPI.
http://www.liberouter.org/vhdl_design/scampi.php.

[6] FlowMon.
http://www.liberouter.org/flowmon/index.php.

[7] NIC.
http://www.liberouter.org/nic/index.php.

[8] NIFIC.
http://www.liberouter.org/nific.php.

[9] Traffic Scanner.
http://www.liberouter.org/ids.php.

[10] NetCOPE.
http://www.liberouter.org/netcope/index.php.

[11] Netopeer project.
http://www.liberouter.org/netopeer/about.php

[12] Netconf.
http://en.wikipedia.org/w/index.php?title=Netconf&oldid=123491664.

[13] NetFlow.
http://en.wikipedia.org/w/index.php?title=Netflow&oldid=117368965.

[14] PHP Manual.
http://cz2.php.net/manual/en/index.php.

[15] Liberouter Wiki.
https://www.liberouter.org/wiki/index.php/Main_Page.