VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATIZOVANÁ TVORBA STATISTIK SÍŤOVÉHO PROVOZU
AUTOMATED CREATION OF STATISTICS OF NETWORK TRAFFIC

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Jan Benedikt**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Číka, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Jan Benedikt ***ID:*** 173609
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Automatizovaná tvorba statistik síťového provozu

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je návrh a realizace modulu pro automatickou tvorbu statistik síťového provozu, které budou mít výstup ve formě reportu. Proveďte analýzu struktur reportů, definujte jejich náležitosti a navrhněte koncept softwarového programu, který by generování reportů realizoval. Na základě návrhu realizujte funkční modul nástroje JMeter, který generuje reporty ze získaných dat. Reporty budou mít podobu interaktivní webové stránky, která využívá technologie HTML, CSS a JavaScript.

DOPORUČENÁ LITERATURA:

[1] GASSTON, Peter. The modern Web: multi-device Web development with HTML5, CSS3, and JavaScript. San Francisco: No Starch Press, 2013. ISBN 15-932-7487-4.

[2] NIEDERST ROBBINS, Jennifer. Learning Web design: a beginner's guide to HTML, CSS, JavaScript, and web graphics. Fourth edition. Beijing: O'Reilly, 2012. ISBN 978-144-9319-274.

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: Ing. Petr Číka, Ph.D.
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Bakalářská práce se zabývá zpracováním a prezentací výsledků zátěžového testování. V první části práce jsou rozvedeny základní pojmy, které jsou úzce spjaté se zátěžovým testováním. Následuje část s analýzou dostupných řešení a jejich vzájemné porovnání. Získané teoretické znalosti jsou následně uplatněny v návrhu modulu pro generování reportu ve formě webové stránky. Samotný modul je vytvořen pro program JMeter a naprogramován v jazyce Java. Vygenerovaná webová stránka je programována v jazyce HTML, CSS a JavaScript.

KLÍČOVÁ SLOVA

Zátěžové testování, statistiky, JMeter, web, Java, JavaScript, HTML, CSS, XML, DDoS

ABSTRACT

This Bachelor Thesis aims to process and present outcomes of stress testing. The first chapter depicts basic terms closely connected to stress testing. Second part is concerned with the analysis of possible resolutions and their comparison between each other. The obtained theoretical knowledge is then applied to model sketch for a report, generated in form of the website. The model itself is created in JMeter program and it is written in the Java coding language. The generated website is programed in HTML, CSS and JavaScript languages.

KEYWORDS

Performance testing, load testing, statistics, JMeter, web, Java, JavaScript, HTML, CSS, XML, DDoS

BENEDIKT, Jan *Automatizovaná tvorba statistik síťového provozu*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 68 s. Vedoucí práce byl Ing. Petr Číka, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Automatizovaná tvorba statistik síťového provozu“ jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Petru Číkovi, Ph.D. za odborné vedení. Dále bych rád poděkoval Haně Lukešové.

Brno ....

podpis autora(-ky)

GiTy, a.s.
Mariánské náměstí 1
617 00 Brno – Komárov
Czech Republic
www.gity.eu

PODĚKOVÁNÍ

Tato bakalářská práce byla realizována ve spolupráci s firmou GiTy a.s., za což této firmě a jejím zaměstnancům panu Ing. Martinu Sýkorovi a panu Ing. Štěpánovi Grabovskému, patří poděkování.

Brno .....

podpis autora(-ky)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM VÝPISŮ

ÚVOD

Při provozování serveru, který poskytuje služby veřejnosti, je v mnoha případech nutné zajistit jeho bezproblémový chod. Nabízí se mnoho typů testování, které jsou schopné odhalit, jak velký provoz je server schopný obsloužit. Jedna z možností je zátěžové testování. Zátěžové testování funguje na stejném principu jako odepření služby, DoS (Denial of service). Tento typ útoku velkým množstvím dotazů zahltí server a ten přestane vyřizovat požadavky klientů. To může znamenat v komerční sféře velký problém, kdy nefunkční internetové služby znamenají velké finanční ztráty. Aby se těmto výpadkům zamezilo, lze pomocí zátěžových testů zjistit slabiny v serverové infrastruktuře.

V první sekci teoretické části této práce bude nastíněn důvod pro zátěžové testování a typy zátěžového testování. V této sekci budou také uvedeny nejdůležitější hodnoty a informace, které zátěžový test dokáže vygenerovat. V druhé kapitole, bude provedena analýza třech dostupných řešení na Internetu. U těchto řešení bude především kladen důraz na podání výsledků. Na stránce s výsledky se bude analyzovat množství podaných výsledků a interaktivnost práce s touto stránkou. Třetí sekcí bude analýza programu JMeter. V tomto programu budou analyzovány moduly, které generují výsledky. Druhá oblast zájmu v programu JMeter se bude týkat generování výsledků testu do souboru XML, kde budou popsány možnosti exportu výsledků do tohoto souboru. V poslední sekci budou porovnána všechna testovaná řešení a bude provedeno závěrečné zhodnocení poznatků, které byly v průběhu práce získány.

Po teoretické části bude následovat část popisu vyhotovení vlastního řešení. V úvodní části budou popsány technologie, které budou následně využity pro vytvoření vlastního modulu pro generování statistik do interaktivní webové stránky. V druhé sekci bude popsán programový kód modulu pro program JMeter. Tento modul bude vytvářen pomocí programovacího jazyka Java. V třetí sekci bude proveden návrh vzhledu webové stránky, která bude výsledkem zátěžového testu pomocí programu JMeter. Dále bude popsán zdrojový kód navržené šablony interaktivní webové stránky. Popisování zdrojového kódu bude rozděleno do dvou podsekcí, kde první podsekce bude popisovat kostru stránky vytvořenou pomocí Hypertextového značkovacího jazyka, HTML (HyperText Markup Language) a kaskádových stylů, CSS (Cascading Style Sheets), ta bude mít název „Vizuální část“. Druhá podsekce bude obsahovat popis výkonné části (kde „Výkonná část“ je i její název“), zpracovávající data výsledků testu. Tato výkonná část bude vytvořena pomocí jazyka JavaScript.

1 ZÁTĚŽOVÉ TESTOVÁNÍ

Mnoho systémů musí podporovat souběžný přístup stovek nebo tisíců uživatelů. Velký růst Internetu přispěl k zvýšené potřebě aplikací, které fungují rychle a v jakoukoliv dobu. Problémy s výkonem serveru jsou často objeveny pozdě. To může znamenat, že náhlý nápor uživatelů na server způsobí jeho selhání. Následné náklady na opravu můžou být vysoké. Aby se předešlo této situaci, je zde možnost využít zátěžového testování [5]. Tento typ testu dokáže odhalit míru návštěvnosti serveru, kterou ještě dokáže obsloužit.

Dalším důvodem použití zátěžového testování je obrana proti DoS útokům. Také lze hovořit o DDoS (Distributed Denial of Service) útocích, kde je útok veden z více míst. Je to typ útoků, které naruší, nebo úplně znepřístupní službu, na kterou útočí. Protože většina firem a organizací je dnes závislá na síťových službách, vzniká při útoku velký problém. I několikaminutový útok může znamenat velké finanční ztráty a ztrátu renomé. Aby firma, instituce či organizace předešla takovým problémům, nabízí se využití zátěžového testování.

Zátěžové testování je ve své podstatě řízený DoS útok, který dokáže zjistit do jakého množství dotazování ze strany klientů je server schopen reagovat a s jakou chybovostí. Dále lze tímto testem zjistit míru efektivity ochrany proti těmto typům útoků a případně její následnou optimalizaci. Zátěžové testování lze uplatnit na mnoho síťových prvků. Tato práce se zabývá testováním webových služeb dostupných v síti Internet. Dalším důležitým síťovým prvkem, který musí být odolný proti vysoké zátěži jsou hardwarové firewally. Lze také testovat celou síťovou infrastrukturu.

Pomocí zátěžového testování lze také odhalit slabá místa testované aplikace. Lze jím odhalit neočekávané chování služby při specifickém úkonu uživatele. Například pokud dva uživatelé zažádají o daný typ služby v jeden čas a tento úkon může přinést problémy, které nelze základním testováním odhalit. Zátěžové testování lze rozdělit do několika typů. Každý typ zátěžového testování se využívá pro jiný účel a probíhá jiným způsobem.

1.1 Typy zátěžových testů

Následný seznam byl převzat z webu [13].

Výkonnostní test (Performance Test): zatížení systému definovanou zátěží pro změření jeho chování. Používá se pro měření reakcí očekávané zátěže, nebo porovnání vlastností systému po provedené změně. Je součástí performance tunningu.

Test hraniční zátěže (Load/Stress Test): zatěžování systému narůstajícím počtem paralelních procesů. Účelem je najít limit, při kterém aplikace překročí akceptovatelné požadavky.

Test odolnosti (Soak Test): jedná se o dlouhodobé testy, které odhalují nedostatky v aplikaci při jejím nepřetržitém provozu.

Test selhání (Failover Test): tímto typem testů je možné ověřit chování systému v případě jeho selhání a nahrazení záložním systémem. Také je možné ověřit rychlost jeho zotavení pokud je pod zátěží proveden restart, nebo přepojen některý ze systémů, se kterým hlavní aplikace komunikuje.

Test části infrastruktury (Targeted Infrastructure Test): test zaměřený na konkrétní úroveň infrastuktury/architektury řešení. Pomocí testu je možné zjistit, ve které části řešení je nejslabší místo. Používá se velice často u složitých systémů.

Test citlivosti sítě (Network Sensitivity Test): jedná se o performance a stress testy, které se nezaměřují na výkonnost systému, ale rychlost a spolehlivost sítě.

Test objemu dat (Volume Test): tento typ testu slouží k ověřování chování aplikace při zvyšujícím se objemu dat (např. dopad rychlosti volání při narůstajícím objemu dat v databázi).

Zátěžové testování lze provádět několika možnými způsoby. Nejčastějším krokem, který potenciální zájemce provede, je analýza dostupných řešení, která již existují a jsou dostupná na Internetu. Výhodou již hotových řešení na Internetu je jejich dostupnost a ve většině případů jednoduché ovládání. Druhá výhoda spočívá ve velkém výpočetním výkonu datacenter, ve kterých služby běží. Tato datacentra jsou fyzicky umístěná po celém světě a je tedy možné simulovat útoky i z nejrůznějších geografických lokalit světa. Nevýhodou však může být vysoká cena za jejich provoz. V následující části textu budou rozebrána nejčastěji využívaná nastavení analyzovaných řešení dostupných na Internetu. Následně bude věnována pozornost nástroji JMeter, který je však nutné, na rozdíl od předchozích nástrojů, nainstalovat a spouštět z vlastního zařízení.

1.2 Parametry zátěžového testování

Při vytváření projektu zátěžového testování je na místě zvolit parametry tak, aby odpovídaly reálné hrozbě útoku. V následujícím seznamu budou vypsány základní parametry při vytváření testu.

Jednotná adresa zdroje, URL (Uniform Resource Locator) – adresa testovaného serveru.

Počet uživatelů – počet uživatelů, kteří budou server dotazovat.

Úvodní fáze (Rampup) – doba, za kterou dosáhne stanovený počet uživatelů plné aktivity.

Délka testu – celková doba trvání testu.

Toto jsou základní možnosti nastavení, které se nachází v každém testu. U analyzovaných řešení se mohou vyskytnout další upřesňující údaje.

Ve výsledcích jsou hlavními parametry:

Vzorky (Samples) – počet odeslaných požadavků na testovaný server.

Čas odezvy (Response time) – čas odpovědi od serveru. Hodnoty můžou být v průměru, maximu, minimu a další.

Kód odpovědi (Response code) – stavové kódy odpovědi. Tyto kódy informují o stavu vyřízení požadavku. Následující seznam obsahuje základní rozdělení:

1xx – informační

2xx – úspěch

3xx – přesměrování

4xx – chyba u klienta

5xx – chyba na straně serveru

Míra chyb (Error rate) – procentuální vyjádření počtu chyb, které nastaly během testu.

Latence (Latency) – doba odezvy [ms]. Hodnoty mohou být opět v průměru, maximu, minimu a dalších hodnotách.

Trvání (Duration) – délka trvání testu.

Počet chyb (Error count) – počet chyb, které během testu nastaly.

Šířka pásma (Bandwidth) – šířka pásma [Mbit/s].

Celkově přijatých (Total received) – celkové množství přijatých dat.

Další údaje, které se vyskytují v menší míře jsou vypsány u konkrétních řešení.

2 ANALÝZA DOSTUPNÝCH NÁSTROJŮ

Hlavní částí zadání této práce je generování interaktivních webových stránek s výsledky testů z programu JMeter. Protože se na Internetu jistá řešení již nachází, nabízí se tato dostupná řešení vyzkoušet a zjistit zaběhlé metody v práci s výsledky výkonových testů. Při hledání těchto nástrojů na Internetu bylo nalezeno množství různých řešení. Nejrozšířenějším typem řešení jsou testovací nástroje pro webové vývojáře, kdy je možné otestovat stránky a nalézt jejich slabiny v realizaci kódu a ty poté na doporučení testovacího nástroje opravit. Ukázku takových nástrojů lze nalézt například zde [8]. Dalším typem jsou nástroje jako je dále rozebíraný Blazemeter a Load Impact. Tyto nástroje cílí na správce serverů, potažmo poskytovatele hostujících služeb pro webové stránky. Jejich hlavním úkolem je právě zátěžové testování.

2.1 Blazemeter

Domovská stránka nástroje Blazemeter a dokumentace k nástroji je dostupná zde [1]. Blazemeter je asi nejznámější nástroj pro zátěžové testy webového serveru na Internetu (podle žebříčku dostupných služeb na Internetu [9]). Tato služba staví na programu JMeter a mnohé funkce z ní přímo vychází. Blazemeter mimo funkcionalit nástroje JMeter nabízí i mnohé další nástroje pro testování. Jedním z těchto nástrojů je například služba Taurus, která je určena pro průběžné testování. Mezi další nabízené služby můžeme například zařadit: „URL/API Test“, „Webdriver test“ a možnost takzvaných „Multi test“. Následující kapitoly se však věnují pouze testování pomocí technologie JMeter. Samotná technologie je popsána v sekci s názvem JMeter.

2.1.1 Nastavení projektu

Jelikož se jedná o placenou službu, nutností je registrace. V nabídce existuje i bezplatná varianta a je zde tedy možnost vyzkoušet Blazemeter bez jakékoliv investice. Začátečníci mohou využít úvodní založení projektu, které pracuje na systému – krok po kroku. V prvním kroku existují dvě možnosti:

1. Vyplnit pouze doménu nebo IP (Internet Protocol) adresu testovaného serveru.
2. Nahrát kompletně nastavený projektový soubor z programu JMeter.

Po vyplnění všech údajů se spustí automatický test.

Pokud je zakládán nový projekt v administraci uživatele, je zde mnohem více parametrů k nastavení. I zde je možnost nahrát již nakonfigurovaný projektový soubor z programu JMeter, který nastaví nabízené parametry za uživatele. První volbou je

zvolení verze programu JMeter a verze programovacího jazyka Java, ve kterém bude test probíhat. Dále se nabízí nastavení několika parametrů testu:

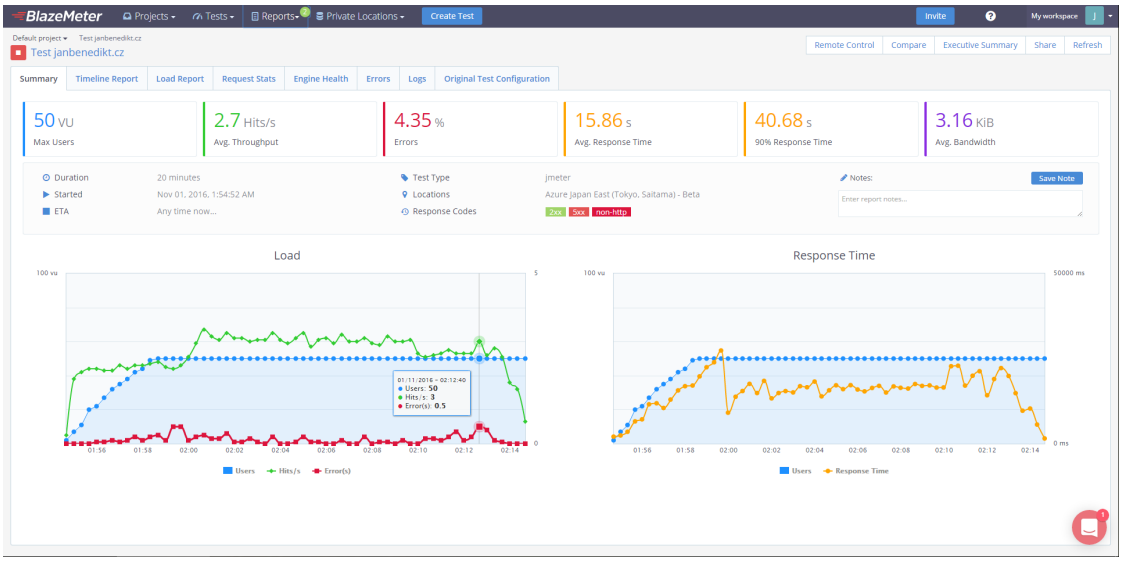
- **Test Failure Criteria** – nastavení kritérií, která mohou nastat při testování (doba odezvy, chyby, velikost přijatých dat, latence, hity). Při nastavování je možnost zohlednit pouze hodnoty mající určitý charakter. Například jsou vyšší, nižší, stejné, nebo se vůbec nerovnají požadované hodnotě. Pokud taková situace nastane, existuje volba test kompletně ukončit.
- **DNS - Hostname Override** – přepis Systému doménových jmen, DNS (Domain name service) hodnot. Je-li potřeba určité doménové jméno přesměrovat na jinou IP nebo vytvořit doménové jméno nové.
- **JMeter Properties** - vytvoření zvolených vlastností pro test v programu JMeter.
- **Command line arguments** – parametry, které lze programu JMeter předat přes příkazovou řádku.
- **Network Emulation** – zde je možnost nastavit simulaci specifické sítě (WiFi, LTE (3GPP Long Term Evolution), 3G (Síť třetí generace), Edge). U mobilních sítí lze zvolit i kvalitu signálu. Z manuálních nastavení je možnost zvolit propustnost stahování a to od 240 do 300000 kB na jedno zařízení/uživatele. Další možností je nastavit latenci, a to v intervalu od 0 do 5000 ms. Poslední možnost nastavení je ztrátovost paketů od 0 do 100 %.
- **CloudWatch** – možnost provázání se službou CloudWatch od firmy Amazon.
- **New Relic** – možnost provázání se službou New Relic.
- **DynaTrace APM** – možnost provázání se službou DynaTrace.
- **AppDynamics** – možnost provázání se službou AppDynamics.
- Dále se zde nachází volba, nechat si zaslat email po dokončení testu.
- Pokud uživatel projeví zájem, je zde možnost exportovat výsledky testu do formátu Hodnot oddělených čárkami, CSV (Comma-separated values).
- **Sandbox Mode** – je-li potřeba, nachází se zde volba spustit takzvaný „režim pískoviště“, umožňující provádět ladění testu.
- Lze zvolit **server**, který bude test provádět. Na výběr jsou tři řešení: Amazon, Microsoft Azure a Google. Dále lokality jako: Spojené Státy Americké, Austrálie, Evropa, atd.
- Důležitá hodnota v testu je **počet dotazujících uživatelů** na testovaný server. V řešení zdarma je možnost si vybrat od 1 uživatele po 50 uživatelů. Pokud je potřeba zvolit jinou metodu dotazů, než je původně zmíněná, je zde volba „Engine“ a počet vláken na jeden „Engine“.
- Pro reálnou simulaci zatížení je možnost zvolit hodnotu **„Rampup“**.
- Předposlední hodnotu, kterou je možno nastavit je **Iterace**.
- A poslední hodnota k nastavení je délka testu. V případě testu zdarma, je

maximální hodnota 20minut.

Tímto je test nastaven. Následuje spuštění testu. Po spuštění Blazemeter oznámí souhrn zvoleného nastavení. Start testu trvá kolem 2–4 minut. Po této době se již zobrazí grafy a informace o výsledcích testu, které jsou v průběhu testu neustále doplňovány o nová data. Spuštěný test při jeho běhu může uživatel kdykoliv vypnout.

2.1.2 Výsledky testu

Další řádky se věnují rozboru výsledků, které Blazemeter generuje. Stránka s výsledky je přehledně členěna do několika sekcí (karet). První karta je věnována shrnutí (viz obr. 2.1). Na této stránce se zobrazují souhrnné výsledky předešlého testu. Na horní části obrázku (stránky) je tabulka s průměrnými hodnotami sledovaných výsledků.



Obr. 2.1: První karta s výpisem základních hodnot.

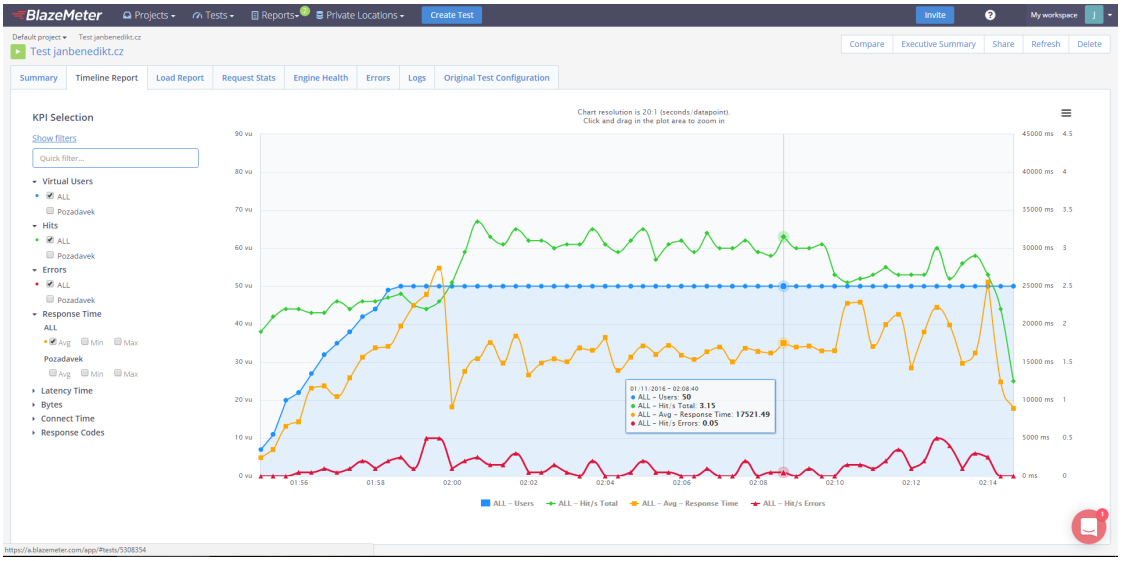
První hodnota představuje maximální počet uživatelů, kteří server dotazovali. V případě testu zdarma se jedná o maximální povolený počet 50 uživatelů. Toto číslo může být prvním ukazatelem úspěšnosti testu. Pokud totiž nastavený počet uživatelů nebude aktivní, může to znamenat špatné nastavení testu nebo zastavení testu kvůli podmínce určené v sekci „Test Failure Criteria“. Dále je v testu nastavená hodnota „Rampup“, která způsobuje postupné spouštění aktivity jednotlivých uživatelů. Druhá hodnota je průměrná propustnost, která zobrazuje počet dotazů na server za 1 sekundu. Třetí hodnota je počet dotazů ukončených chybou. Chyby jsou vyjádřeny pomocí stavových Hypertext transportní protokol, HTTP (Hypertext Transfer Protocol) kódů. Čtvrtou hodnotou je průměrný čas, za který přišla

od testovaného serveru odpověď. Další hodnota znázorňuje čas, který nastal v 90 % případů. Poslední hodnota je průměrná hodnota velikosti přijaté odpovědi.

V další části souhrnu jsou zobrazeny informace o testu: délka trvání testu, datum a čas začátku testu, datum a čas konce testu, typ testu, umístění serveru provádějící test a poslední informací jsou obdržené stavové HTTP kódy.

Poslední část souhrnu vyplňují dva grafy. První graf obsahuje informace o zatížení. V grafu jsou vykresleny hodnoty počtu uživatelů, počtu dotazů a počtu chyb v závislosti na uplynulém čase. V základním nastavení vykreslení grafu je zobrazena pouze závislost počtu uživatelů. Druhý graf obsahuje informace o době odezvy. V grafu jsou zobrazeny hodnoty informující o počtu dotazujících uživatelů a odezvě dotazovaného serveru, stejně jako u prvního grafu v závislosti na čase. Každou hodnotu v grafech je možno skrýt a znovu zobrazit. Grafy jsou interaktivní a při umístění kurzoru myši zobrazí hodnoty příslušné k času, u kterého se kurzor nachází.

V druhé kartě (viz obr. 2.2) je vykreslen graf, do kterého je možnost vykreslit všechny hodnoty, které je možno při testu získat. V základní konfiguraci obsahuje hodnoty vykreslené v grafech, v první kartě. V postranní nabídce jsou příslušné volby pro skrytí a zobrazení jiných hodnot v grafu. Jedná se o hodnoty – čas odezvy, bajty za sekundu, doba připojení a kódy odpovědi. U každé hodnoty lze zvolit 3 módy zobrazení: průměr, minimum a maximum z vybraných hodnot. Graf je možné stáhnout z webu a uložit v několika formátech (JPG, PNG, PDF nebo SVG).



Obr. 2.2: Druhá karta se souhrnným grafem.

Další karta obsahuje graf znázorňující Ukazatele výkonnosti, KPI (Key Performance Indicator). Zde opět je možnost zobrazit již zmíněné uživatele, čas odpovědi, latenci, propustnost, počet dotazů za sekundu a počet chyb.

Čtvrtá záložka obsahuje souhrnnou tabulku obsahující všechny naměřené hodnoty, buď v jejich průměrných nebo maximálních hodnotách. V následujícím seznamu jsou uvedeny všechny hodnoty, které lze v tabulce zobrazit. Ukázka této karty je vyobrazena na obr. 2.3. V tabulce jsou hodnoty popsané v úvodní kapitole (samples, response time, latency, bandwidth a error rate).

Avg. Hits/s – průměrný počet odeslaných požadavků na server za jednu sekundu.

90 %|95 %|99 % line (ms) – číselný údaj v záhlaví sloupce udává percentil. Např. 90 % vzorků bylo menších než nově nastalý vzorek a časová hodnota vyjadřuje dobu, která uběhla, než tato situace nastala.

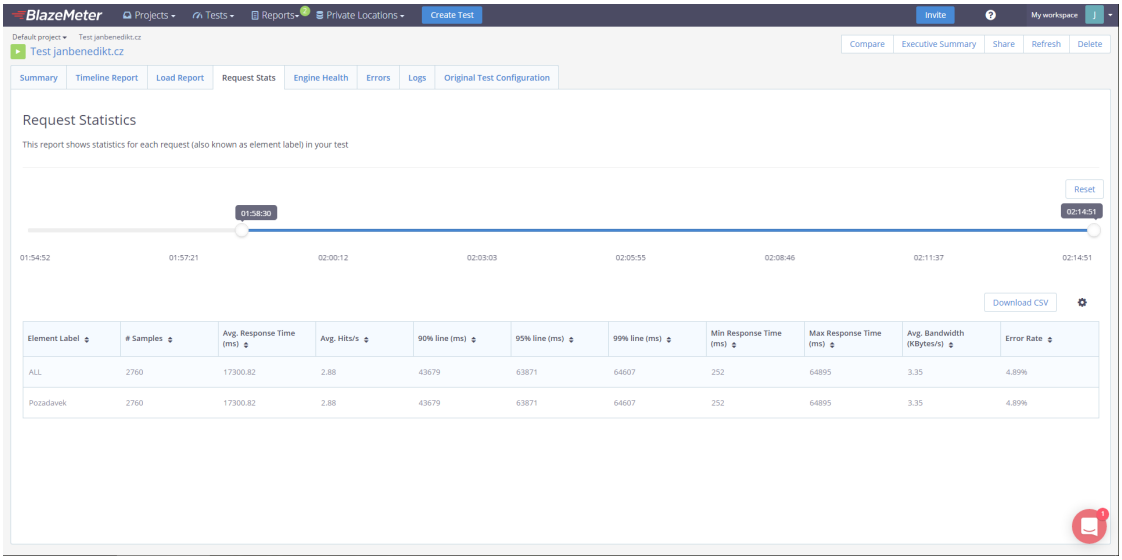
Geo Mean Response Time (ms) – geometrický průměr trvání odpovědi testovaného serveru.

StDev (ms) – standard Deviation – směrodatná odchylka v milisekundách.

Error Count – celkový počet chyb, které nastaly během testu.

Duration (hh:mm:ss) – délka trvání celého testu.

Median Response Time (ms) – medián trvání doby odpovědi testovaného serveru.

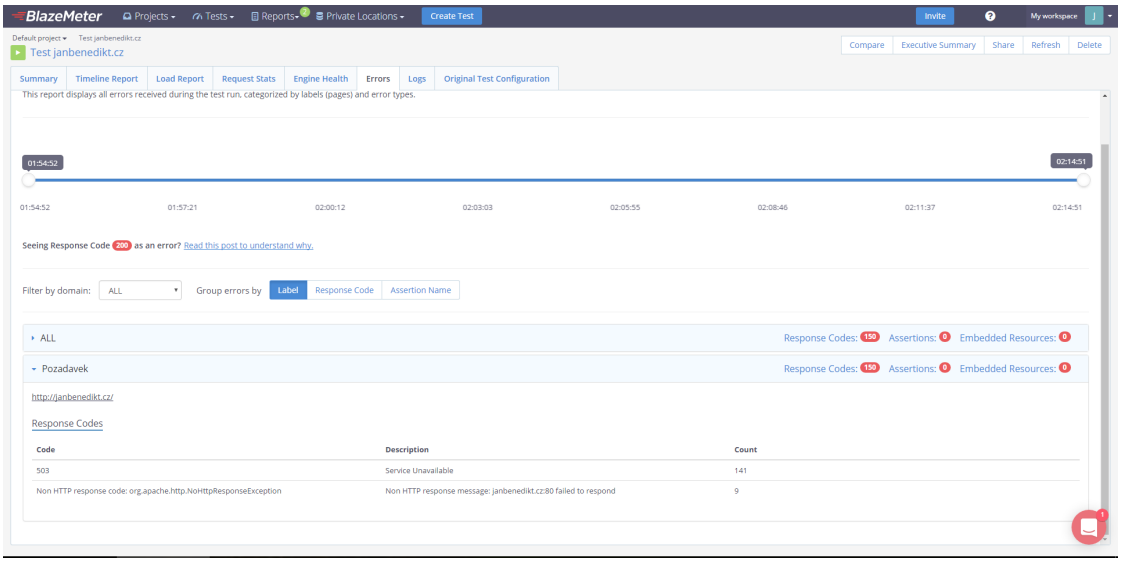


Obr. 2.3: Čtvrtá karta s tabulkou výsledků.

Pátá karta je věnována zdraví testovaného systému (Engine Health). Na této kartě je umístěn graf s informacemi o zatížení systému, který test prováděl. Graf obsahuje data o využití síťového připojení, obsazení paměti, vytížení centrální procesorové jednotky, CPU (Central processing unit) a o počtu připojení.

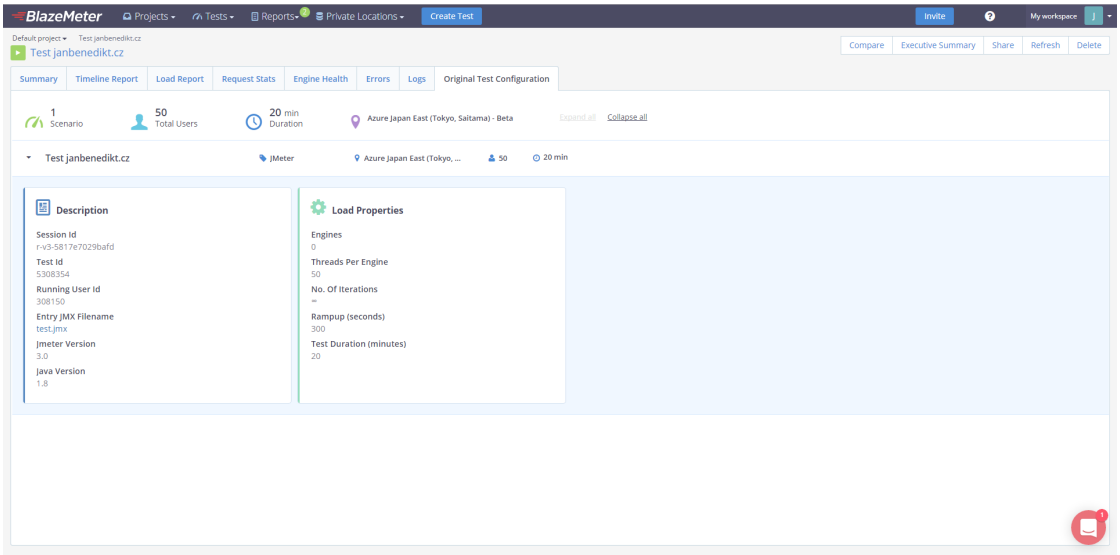
V kartě šesté jsou obsažena data o chybách zaznamenaných během testu. Na horní části karty se nachází časová osa, kde je možnost zvolit časový interval testu

pro přesnější určení příčiny chyby. Chyby lze v základu třídit podle testovaných serverů. Druhou možností je třídění podle názvu obdrženého HTTP stavového kódu anebo podle systémových chyb, které nastaly v programu JMeter. V záhlaví tabulky se nachází součty možných chyb, které byly popsány v možnostech třídění. Po rozkliknutí tabulky se zobrazí samotné sumarizované typy chyb, které během testu nastaly. Sumarizované typy chyb lze vidět v obr. 2.4. Protože není potřeba stejné chyby vypisovat zvlášť, Blazemeter stejné typy chyb sumarizuje a pouze na konci řádku napíše jejich celkový počet. Jako první v řádku chyby je zobrazen kód chyby, následuje popis chyby a dále počet výskytu této chyby. Jak již bylo napsáno, pro přesnější identifikaci výskytu chyby je možné použít časovou osu v horní části karty.



Obr. 2.4: Šestá karta s výpisem chyb, které nastaly během testu.

Jelikož ne vždy je možné vyčíst všechny důležité informace z interaktivních grafů a tabulek, je možné si v další kartě projít kompletní log. Logů je zde k nahlédnutí více. První je vygenerovaný z programu JMeter. Další se týkají podpůrných nástrojů na straně serverů služby Blazemeter. Sekce se rozděluje na další pod karty, kde každá odpovídá určitému logu. Na kartě je název logu a umístění serveru, na kterém program pracoval. V kartě se nachází informace o názvu souboru, session id, typ serveru (konzole), počet řádků v logu a souhrn typů zpráv v logu (informační, varovné, fatální chyby, atd..). U každého logu je možné soubor stáhnout do počítače. Poslední karta umožňuje nahlédnout do logu v prohlížeči (viz obr. 2.5). Zde se dají zprávy vyfiltrovat podle úrovně, která zprávu do logu zapsala. Zde můžeme nalézt celkem tři úrovně – Systém, konzole a uživatel. Dále se dá zvolit Session id, které je potřeba zobrazit. Poslední možností je filtrování zpráv podle jejich typu – oznámení, varování, ladění, informování, chyba a kritická zpráva.



Obr. 2.5: Osmá karta s informacemi o testu.

Poslední karta je souhrnem konfigurace testu. Pro názornost je nad odstavcem obr. 2.5. Jsou zde zobrazeny informace o počtu scénářů, nastaveném počtu uživatelů, trvání testu a umístění serveru, na kterém test běží. Níže jsou údaje o použitém testovacím prostředí (v našem případě JMeter). Dále se na kartě nachází dvě tabulky, které zobrazují informace o Session ID (identifikační číslo výsledků daného testu – je zde možnost mít na jeden test více výsledků), identifikační číslo testu, identifikační číslo uživatele, název vloženého projektu JMeter a verzi JMeteru s verzí Javy.

Poslední položka na stránce s výsledky je menu, které obsahuje funkci pro porovnání výsledků mezi sebou. Je možné vygenerovat stránku se základními výsledky testu a také je možnost výsledky sdílet ostatním uživatelům. Pokud je potřeba, existuje zde i tlačítko pro obnovu stránky a smazání výsledků testu.

2.2 Load Impact

Domovská stránka nástroje Load Impact a dokumentace k nástroji je dostupná zde [15]. Dalším řešením, které je na Internetu dostupné je testovací nástroj Load Impact. Řešení Load Impact nestaví na základu JMeter, jako Blazemeter, ale využívá proprietární řešení, jež nebylo možné blíže prozkoumat. Load Impact nabízí podobné možnosti jako Blazemeter, avšak ve výsledcích je značně omezený. Protože i Load Impact je placený nástroj, byl využit pouze úvodní test zdarma. Při vytvoření nového účtu je uživateli automaticky přičteno na jeho účet určitý počet kreditů. Kredity zde slouží jako virtuální měna, kterou se platí jednotlivé úkony v testech. Pro uživatele, který chce službu pouze bezplatně vyzkoušet to může způsobit jistá

omezení. Například nemůže využít dlouhodobé testovaní, které vyžaduje více kreditů, než má uživatel v základní verzi zdarma k dispozici. Po registraci a následném přihlášení se zpřístupní uživatelská administrace.

Uživatelská administrace se dá rozdělit do několika sekcí. V horní části lze zvolit jeden z již existujících projektů. Po zvolení projektu se zobrazí textové pole pro vyplnění webové adresy nebo IP adresy. Pod tímto textovým polem jsou další dvě textová pole, první kolonka je nastavení počtu uživatelů a druhá je pro zvolení délky testu v minutách. Pokud je potřeba zvolit více parametrů, existuje volba „Run a more realistic test, start here“. Dále se stránka dělí v menu do následujících sekcí:

Tests – souhrnný seznam všech testů ve zvoleném projektu.

Users scenarios – zde jsou uživatelské scénáře. Uživatelský scénář definuje, jak se bude testovací server dotazovat testovaného serveru.

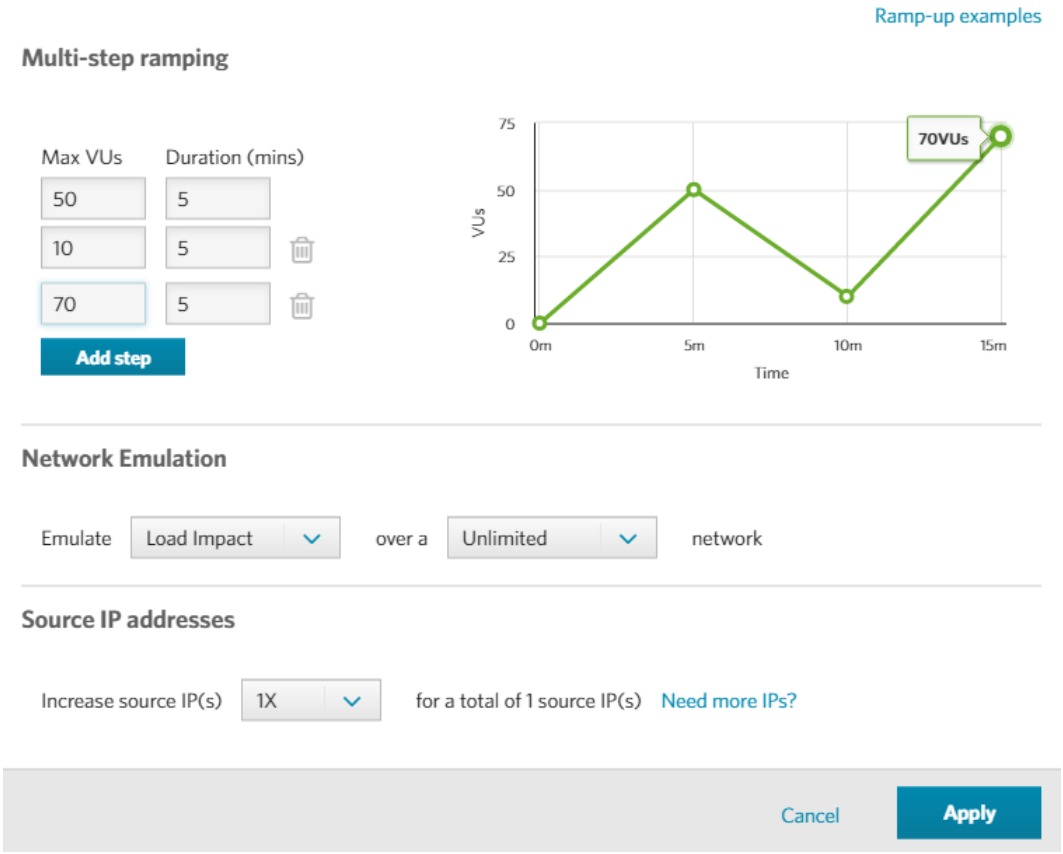
Data stores – datové sklady slouží pro uložení informací potřebných pro zpřístupnění testovaného webu. Lze zde uložit například přihlašovací informace, jako přihlašovací jméno a heslo.

Monitoring – v placené verzi nástroje lze v této sekci zvolit průběžné monitorování testovaného serveru.

Integrations – tato sekce obsahuje několik služeb, do kterých lze Load Impact integrovat, tzn. využít ho ve složitější cloudové struktuře.

2.2.1 Založení projektu

Z výše uvedeného textu vyplývá, že pro založení testu lze zvolit dvě cesty. První cesta je pouze zvolení testované webové, nebo IP adresy. Tato metoda je nejrychlejší, avšak nemusí být nejefektivnější. Druhá cesta je komplexnější nastavení projektu. Zde se jako první vyplňuje adresa testovaného serveru. Po potvrzení zadané adresy pokračuje průvodce konfigurací na další stránku, kde lze nastavit fyzické místo testovacího serveru, který bude test vykonávat. Na výběr je lokací jako například: Ashburn (USA), Dublin (Irsko), Palo Alto (USA), Singapur (Asie), Sydney (Austrálie) a také Tokio (Japonsko). Následuje nastavení počtu uživatelů a délky testu. Oproti řešení Blazemeter je zde výhoda v nastavení nárustu aktivity dotazujících uživatelů. V případě Load Impact lze počet aktivních uživatelů měnit s postupem času. Jak lze vidět na obr. 2.6, uživatelé zprvu narostou do počtu 50, následně se počet zredukuje na 10 a poté vzroste na 70. V tomto dialogu, který je znázorněn na obr. 2.6, lze také nastavit simulaci webového prohlížeče (Internet Explorer, Google Chrome, aj..), použitou síť (LTE, 3G, Edge, aj..). Poslední možností v tomto dialogovém okně je nastavení aktivních IP adres (tato možnost je však dostupná pouze pro placené účty).



Obr. 2.6: Nastavení počtu uživatelů v testu.

Po odsouhlasení dialogového okna s nastavením je dalším krokem nastavení prahových hodnot testu (obdoba „Test Failure Criteria“ u služby Blazemeter). Zde je možné nastavit tyto hodnoty:

Failure rate – počet dotazů na server, které skončí nezdarem.

VU load time – délka času odpovědi testovaného serveru v milisekundách.

Custom metric – zde si lze zvolit vlastní sledované hodnoty. Tyto hodnoty se definují pomocí skriptu uživatele.

Stejně jako u stránky Blazemeter i zde je možnost po dosažení sledovaných hodnot test ukončit. Následující položka nastavení je pro monitorování prostředků testovaného serveru. Protože velký nápor uživatelů webu může mít za následek přetížení serveru, je zde metoda jak jednoduše zjistit slabé místo výkonnosti serveru. Služba Load Impact obsahuje takzvaný „Server Monitoring Agent“, což je klient, který se nainstaluje na testovaný webový server a propojí se s projektem, který uživatel vytváří. Je zde podpora jak Linuxových distribucí, tak serveru od firmy Microsoft.

Poslední položkou v nastavení testu je sdružování webových adres. Tuto funkci můžeme zavést z důvodů více webových stránek na jednom fyzickém serveru. Pokud

je na jednom webovém serveru více webů a uživatel testuje pouze jednu adresu, nemusí mít výsledek testu takovou vypovídající hodnotu o zatížení, jako když testuje více adres najednou. K tomuto účelu slouží jejich sdružení. Toto sdružení je možné v rámci jednoho testu a poskytne nám tak relevantnější výsledky.

2.2.2 Výsledky testu

Po nastavení a úspěšném spuštění testu se zobrazí první výsledky. Stejně jako u předchozího nástroje Blazemeter, jsou výsledky generovány postupně během testu. Následující hodnoty lze vidět na obr. 2.7. V záhlaví výsledků je název testu a jeho stav (dokončený, v procesu, ukončen chybou), možnost sdílení výsledků, exportování výsledků do CSV a možnost znovu-spuštění testu. Pod touto sekcí se nachází základní údaje o testu. První údaj informuje uživatele o celkových přenesených datech a počtu požadavků. Následují informace o čase začátku testu a jeho ukončení. Vedle informací o čase se nacházejí dvě hodnoty, které informují o počtu nastavených uživatelů a délce testu. Horní část stránky s výsledky vyplňuje sumarizační tabulka. První položka v tabulce je počet aktivních uživatelů, následuje datový tok, počet aktivních TCP (Transmission Control Protocol) spojení, velikost obdržených dat, počet odeslaných dotazů na testovaný server a poslední položka v tabulce je počet odeslaných požadavků během jedné sekundy. Tato tabulka je zobrazena pouze při běžícím testu. Po ukončení testu tabulka zmizí, což může být nepraktické z důvodu pracného dohledávání požadovaných hodnot.

Pod sumarizační tabulkou je mapa, kde je vyobrazeno umístění dotazujícího serveru a serveru dotazovaného. Vedle mapy se nachází jednoduchý log událostí nastalých při testu.

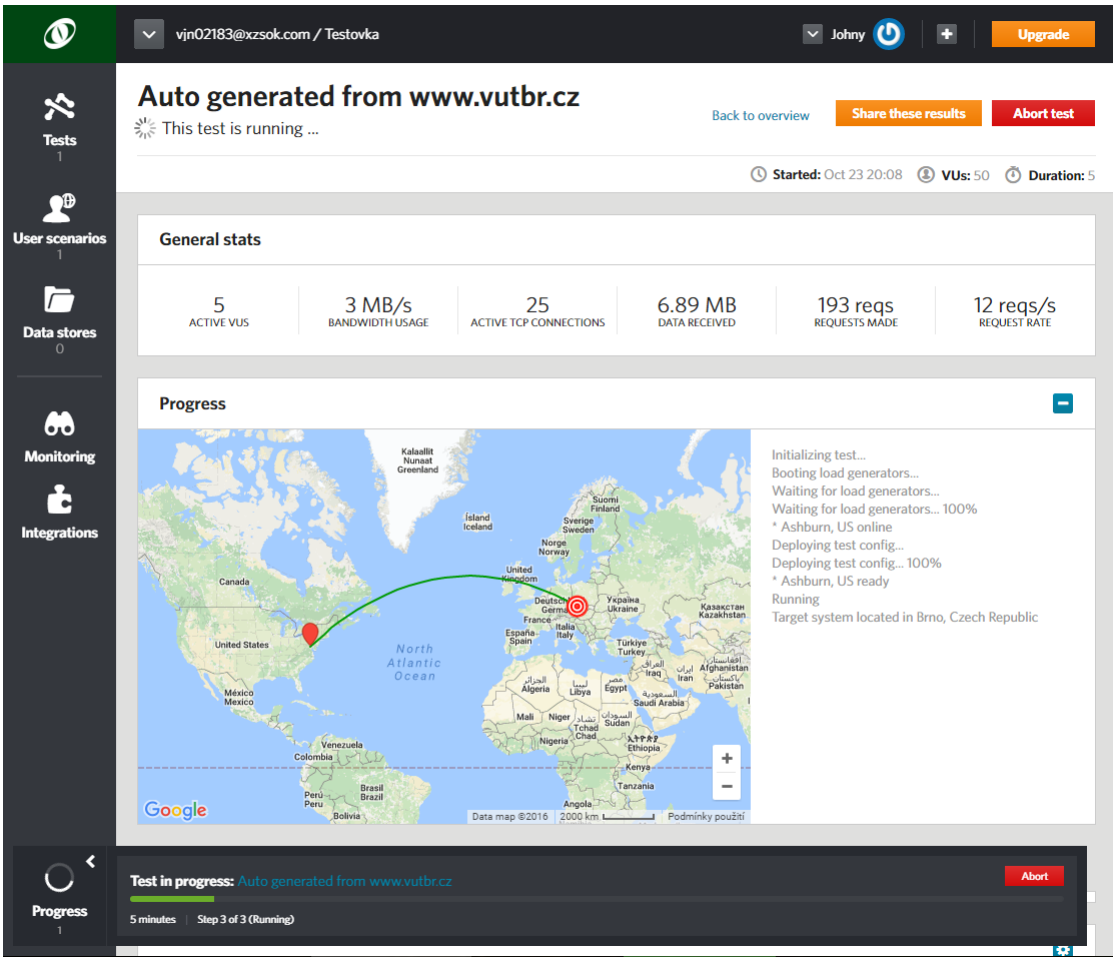
Pod mapou se nachází čtyři karty s výsledky. Na první kartě jsou zobrazeny grafy s výsledky testu (viz obr. 2.8). Na hlavním grafu jsou vykresleny dvě hodnoty. První hodnota je počet aktivních dotazujících uživatelů. Druhá hodnota reprezentuje dobu zatížení uživateli. Do tohoto grafu lze přidat další hodnoty přetažením menšího grafu, který se nachází pod hlavním grafem. Další důležitou věcí, která usnadní práci s hlavním grafem, je časová osa, kterou lze určovat zobrazený časový interval na grafu.

Jak již bylo zmíněno, pod hlavním grafem se nachází další, menší grafy. Mezi grafy lze zobrazit i hodnoty popsané v úvodní kapitole (samples, response time, latency, bandwidth, error rate a total received). Zde je výčet všech možných, které lze zobrazit:

Acc. load time – celkový součet všech časů zatížení.

Delta requests – celkový počet žádostí na testovaný server.

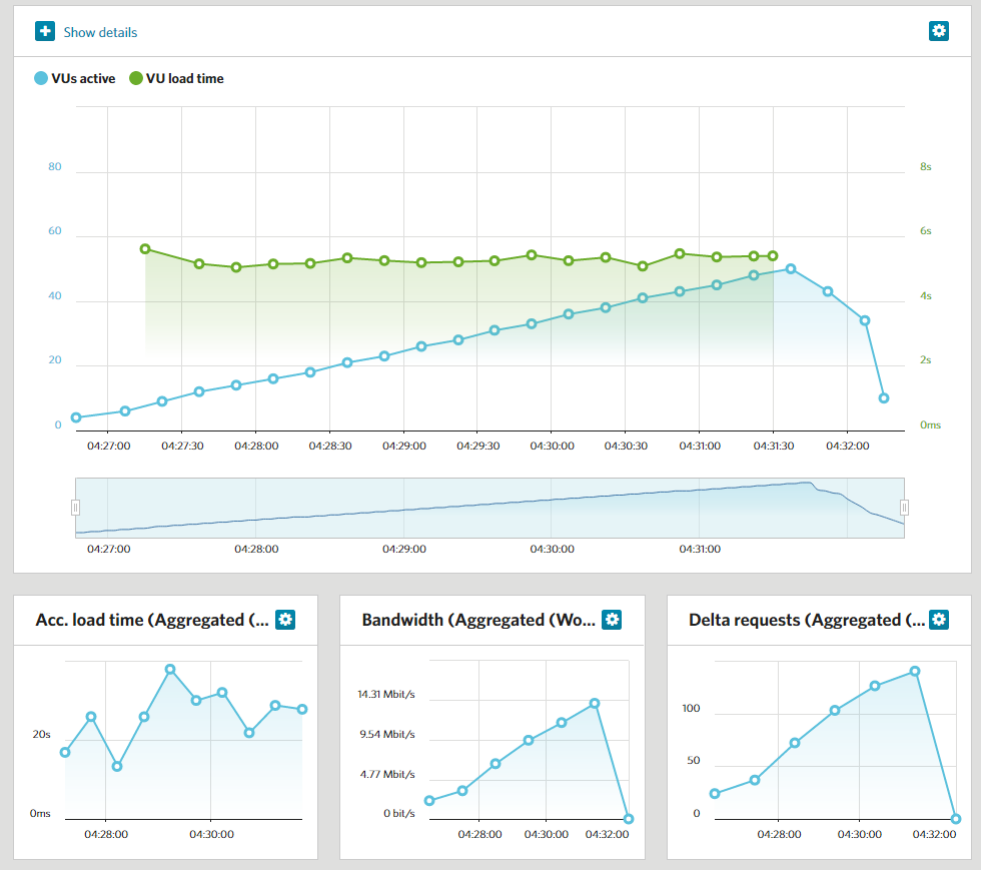
Load generator CPU – zatížení CPU testujícího serveru v %.



Obr. 2.7: Průběžné výsledky během testu.

Load generator memory – využití paměti testujícího serveru v %.
Progress – pokrok v testu, který znázorňuje procentuální vyjádření postupu.
Repetitions failed – procentuálně vyjádřená hodnota opakování dotazů, které skončily chybou. V případě úspěšného testu je hodnota vždy 0 %.
Repetitions succeded – procentuálně vyjádřená hodnota opakování dotazů, které skončily úspěchem. V případě úspěšného testu je hodnota vždy 100 %.
Request/second – počet dotazů na testovaný server za jednotku sekundy.
TCP connections – počet aktivních TCP spojení.
Total received – celkový počet přijatých dat v MiB.
Total requests – celkový počet žádostí na testovaný server.
VU Load time – doba zatížení uživatelů.
VUs active – počet aktivních uživatelů.
Všechny tyto hodnoty jsou vynášeny do grafu v závislosti na čase.

Další karta ve výsledcích obsahuje webové adresy všech částí stránky, které jsou



Obr. 2.8: Část stránky s grafy výsledků testu.

během načítání stahovány. Tyto soubory mohou být například typu CSS, obrázky, JavaScript, HTML soubory a jiné. Hodnoty jsou přehledně zobrazeny v tabulce. První sloupec tabulky je adresa stahovaného souboru. Druhý sloupec je umístění testovacího serveru. Další sloupec obsahuje název vygenerovaného scénáře. Čtvrtý a pátý sloupec obsahuje hodnoty o úspěšném a neúspěšném načtení souboru. Poslední sloupec poskytuje průměrný čas stažení požadovaného souboru.

Předposlední karta poskytuje informace o načítané webové stránce. První údaj informuje o automaticky vygenerovaném názvu pro stránku v rámci testu. Následuje údaj o poloze testovacího serveru. Další údaj je název scénáře. Tabulka pokračuje údaji o celkovém počtu načtení stránky, minimálním, průměrném a maximálním čase načtení stránky.

Poslední karta obsahuje log vygenerovaný během testu. Log se dá filtrovat podle typu události (chyba, informace, ladění, nebo vše). V logu je možné hledat a tak se v něm dá lépe orientovat, hledáme-li konkrétní událost. Log je přehledně vypisován do tabulky, kde první sloupec je časové razítko, za ním následuje informace o umístění testovacího serveru, název scénáře, úroveň události a zpráva události.

2.3 Apica

Domovská stránka nástroje Apica a dokumentace k nástroji je dostupná zde [14]. Posledním komerčním řešením, které je v této práci popsáno, je stránka Apica. Použitá technologie je podobně jako u řešení LoadImpact neznámá. Stejně jako u dvou předchozích řešení, je i Apica placeným nástrojem. U tohoto řešení je místo systému s kredity použito časové omezení v používání všech nabízených funkcí. Takzvaný „trial“ – pokusný účet má veškerou funkčnost placeného, plnohodnotného účtu, avšak jak již bylo zmíněno – s časovým omezením. Po vypršení přiděleného časového období pro vyzkoušení, je uživateli nabídnuta možnost přejít na placený účet. Prvním krokem je tedy založení účtu. Po úspěšném založení účtu se uživateli objeví administrace jeho účtu.

2.3.1 Nastavení projektu

Stránka s administrací má několik sekcí, které jsou následně popsány. V první sekci je souhrnná tabulka s již založenými projekty. Zde je možné projekt založit. Při zakládání projektu je povinné vyplnit pouze jeho jméno. Pro rozlišení mezi více projekty se zde nachází položka pro vyplnění popisu projektu a cílů, které chce uživatel tímto projektem dosáhnout. Pro úplnost je zde možnost vyplnit jméno vedoucího projektu, datum začátku, konce (pokud například je projekt ukončen s úmyslem budoucího pokračování) a jeho úplného konce. Další položky se týkají jeho stavu (zda-li je započat, v průběhu, nebo dokončen). Také se tato hodnota dá nastavit procentuálním vyjádřením jeho stavu.

Po založení projektu se založený projekt zobrazí v tabulce projektů. Tabulka obsahuje základní informace o projektu, jako je například datum startu, poslední provedený test a celkový počet provedených testů. Je-li projekt založen, lze v projektu vytvořit první test.

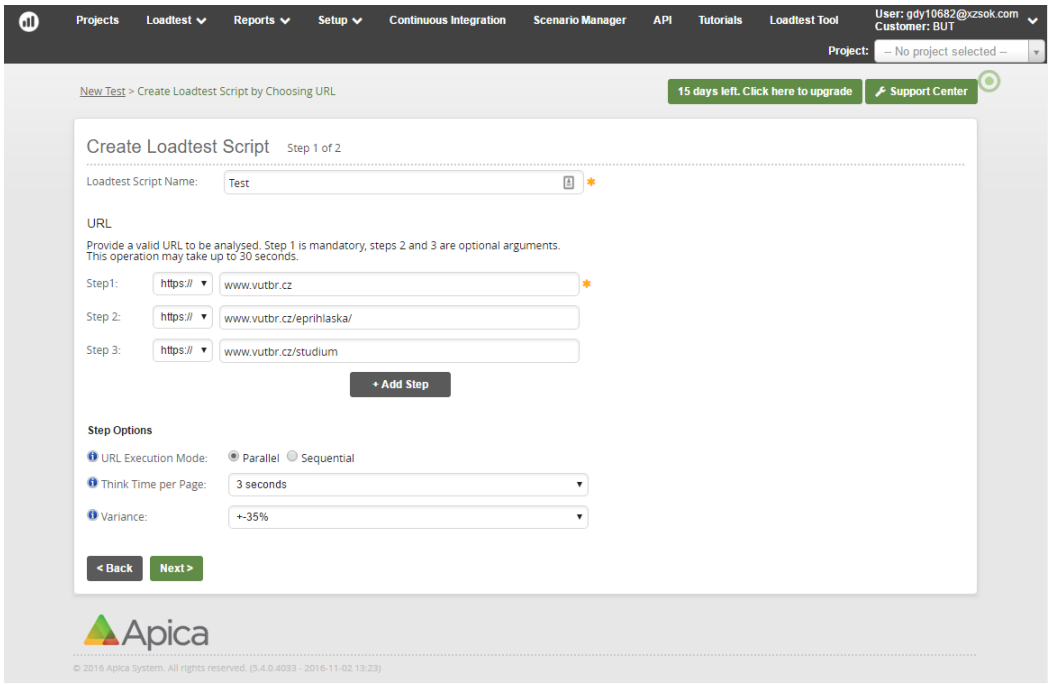
Nový test se zakládá v další sekci stránky. Sekce Loadtest obsahuje jak vytvoření nového testu, tak přehled běžících testů. Vytváření nového testu má logickou posloupnost a stránka uživatele provádí potřebnými úkony k vytvoření funkčního testu. První údaj, který uživatel volí, je typ předplatného, které je svázáno s jeho účtem. Následuje krok, kdy je na výběr nahrání již existující konfigurace, kterou uživatel vytvořil v dřívější době. Pokud taková již existující konfigurace k dispozici není, pokračuje se vytvářením testu. Zde jsou na výběr tři možnosti.

URL – vytvoření tzv. skriptu s několika webovými adresami a nastavením, které říká, jak s nimi bude testovací server zacházet viz obr. 2.9.

Selenium – nahrání konfigurace z programu Selenium.

Zebra Tester – nahrání konfigurace vytvořenou v programu Zebra Tester, který

je vyvíjen společností Apica. Program Zebra Tester také slouží jako klient pro testovaný server, který monitoruje zatížení CPU a paměti RAM (Random-access Memory).



Obr. 2.9: Nastavení skriptu testování serveru.

Při nastavování skriptu, jak je vidět na obr. 2.9, se vyplňují následující náležitosti. Prvním krokem je jeho pojmenování. V dalším kroku se vyplňují domény, na které se budou zasílat požadavky. Stejně jako u služby Load Impact je zde doporučeno vyplnit více adres, které odkazují sice na jeden fyzický server, avšak na jeho různé části. Tím se opět lépe nasimuluje zátěž, která může být na server vyvinuta. Poslední tři položky jsou ukazatele, jak skript bude s vyplněnými adresami pracovat.

URL Execution Mode – testující server bude testovat zadané domény buď paralelně (všechny ve stejný čas), nebo sériově (v jeden čas pouze jednu doménu a bude je postupně střídat).

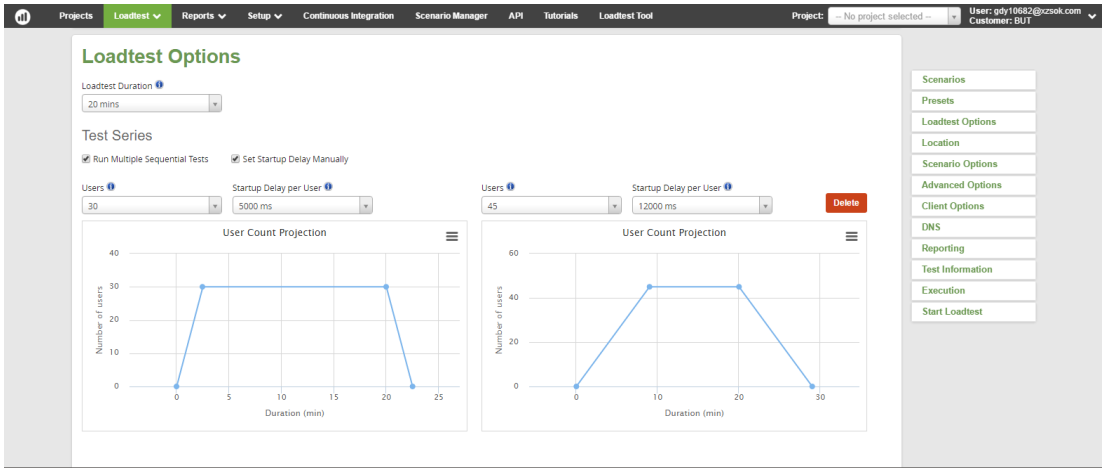
Think Time per Page – čas strávený na testované stránce při jednom načtení.

Variance – povolená odchylka zadaného času.

Po vyplnění všech požadovaných polí testovací server provede načtení webových stránek, které byly zadány. Všechny stažené soubory se zobrazí v tabulce. Každá zadaná stránka má vlastní tabulku a je možné se mezi nimi přepínat v horní části stránky. V prvním sloupci tabulky je číselné označení jednotlivého dotazu na stránku. Následuje sloupec s úplnou adresou stahované položky ze serveru. Třetí sloupec určuje dobu od spuštění načítání stránky, kdy byla položka stažena. V dalším sloupci je

velikost stahované položky v bajtech. Následuje čas v milisekundách, který uběhl při stahování dané položky. Ve sloupci „type“ je zobrazen typ položky (CSS soubor, obrázek, skript, atd..). Následuje sloupec s typem požadavku (GET, POST). Předposlední sloupec obsahuje výsledek dotazu určený stavovou hláškou. Poslední sloupec obsahuje stavový HTTP kód. Tato tabulka slouží pro zvolení položek, které jsou žádoucí při testování a budou stahovány. Po dokončení požadovaných úprav je nastavení skriptu hotové a pokračuje nastavení testu.

Jako u skriptu, i u nastavení údajů v testu je možné zvolit nastavení z předchozích testů. Nastavení testu pokračuje volbou počtu uživatelů, postupného nárustu jejich aktivity a celkového času testu. Oproti předchozím dvěma řešením, je možné zpoždění nárůstu aktivity uživatelů zvolit dvěma způsoby. První způsob je čas, který bude trvat, než se zaktivují všichni uživatelé (zde je zcela na testovacím serveru, v jakých intervalech bude uživatele aktivovat). Druhým způsobem je přesné určení intervalu, mezi aktivováním prvního a následujícího uživatele. Další možností, která není dostupná u předešlých dvou řešení, je vytvoření druhé, zcela nezávislé skupiny uživatelů. Těchto skupin lze vytvořit více, ne jen třeba dvě. Každá skupina má individuální nastavení a může se chovat nezávisle na předchozí „skupině“. Jak lze vidět na obr. 2.10, stránka vykresluje přehledné grafy znázorňující aktivitu uživatelů v průběhu času testu.



Obr. 2.10: Nastavení počtu uživatelů v různých skupinách.

Nastavení testu pokračuje výběrem polohy testovacího serveru. Protože všechna zmíněná řešení využívají stejné poskytovatele datových center, umístění ve světě je stejné jako u předchozích řešení. Vybraný server je možné uložit jako „cluster“ pro pozdější a snadnější výběr. Po výběru lokace následuje možnost zvolit předdefinovaný scénář. Tyto scénáře odpovídají následujícímu nastavení v dalších krocích.

V Advanced Options lze zvolit v první možnosti počet smyček na jednoho uživatele. Tyto smyčky definují, kolikrát se uživatel během testu serveru dotáže na dříve zadané parametry. Následuje hodnota Request Timeout v sekundách. Tato hodnota je doba, kterou je uživatel schopen čekat na odpověď od testovaného serveru. Pokud do této doby odpověď nedostane, dotaz automaticky skončí chybou. Dále je zde možnost přidat dodatečné parametry. Tyto parametry zajistí například každému uživateli unikátní IP adresu, při každém dotazu se naváže nová vrstva bezpečných socketů, SSL (Secure Sockets Layer) relace s testovaným serverem nebo nastaví uživateli jiné časové pásmo. Poslední možností v Advanced Options je možnost rozdistribuování zátěže uživatelů po všech dostupných datacentrech ve zvolené lokalitě.

V dalším nastavení, které má název Client Options lze nastavit simulaci webového prohlížeče, který uživatel používá (Internet Explorer, Google Chrome, Firefox). Ve výběru lze zvolit i mobilní operační systém (iOS, Android, nebo Windows Phone). Následuje nastavení rychlosti odesílání na server a stahování ze serveru (obojí v Mbit/s). Poslední možností je monitorování chování klientů na testovaném serveru. To lze zajistit pomocí výše zmíněného programu Selenium.

Jako u předchozích řešení, i zde lze zvolit vlastní konfiguraci DNS serveru. Tato konfigurace je stejná jako u předchozích dvou řešení. Po nastavení vlastních hodnot u DNS následuje možnost zaslání výsledků testu na email. V případě řešení Apica je možnost zaslání i předešlých výsledků pro porovnání s aktuálním testem.

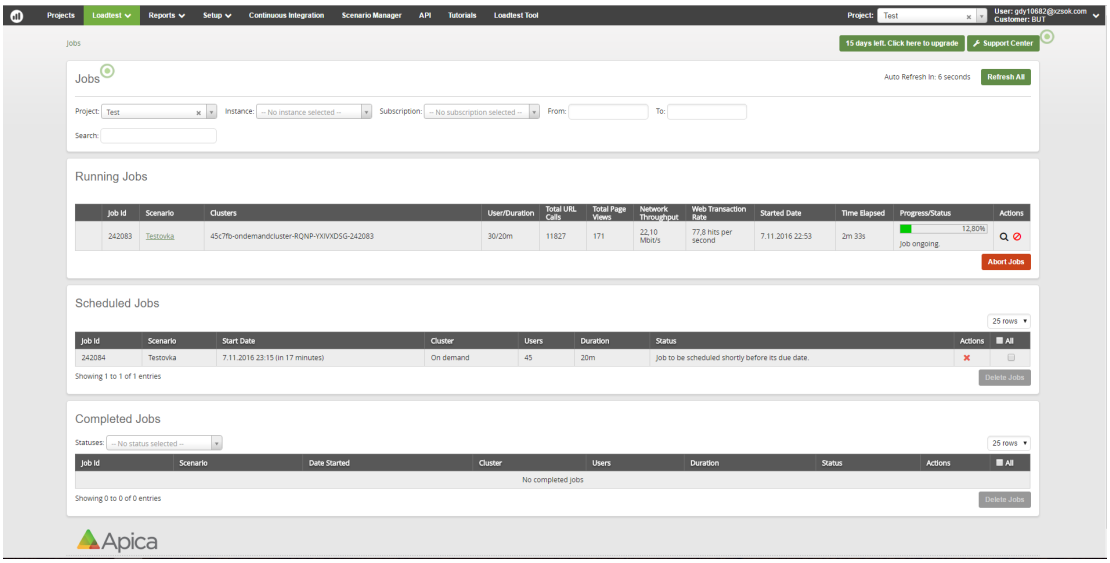
I zde se nachází textové pole pro poznámky k testu a pro tagy (značky), které rozliší jednotlivé testy.

Poslední možné nastavení testu je možnost jeho automatického opakování. Jsou zde tři možnosti. Opakovat týdně, denně a nebo pouze v daný den. Při opakování týdně se nastaví během kolika následujících týdnů se má test vykonávat. Po nastavení všech výše zmíněných hodnot lze test spustit.

2.3.2 Průběžné výsledky testu

Po spuštění testu se zobrazí stránka s tabulkou běžících testů, testů přerušených a testů dokončených. Tuto stránku lze vidět na obr. 2.11.

Při kliknutí na název jednoho z testů se zobrazí stránka s výsledky testu. Na hlavní stránce s výsledky se v záhlaví nachází stav testu. Pod touto informací jsou umístěny informace jako: název testu, jeho procentuálně vyjádřený průběh, počet aktivních uživatelů, celkový počet smyček a možnost přerušení testu. Pod těmito informacemi se nachází pás karet, které rozdělují celou stránku s výsledky do tematických sekcí.



Obr. 2.11: Stránka se souhrnem testů.

První sekce „Shrnutí“ obsahuje základní data a grafy. V horní části se nachází tabulka. První hodnotou tabulky je počet smyček – celkový, úspěšných a neúspěšných. Další hodnotou je počet dotázaných URL adres – dotázaných s úspěchem a ukončené chybou. Následuje celkový počet zobrazení stránky, kterou jsme testovali. Další hodnota v pořadí, je šířka pásma uváděná v Mbit/s. Výsledky pokračují hodnotou aktuálního počtu dotazů na server během jedné sekundy. Protože se jedná o průběžné výsledky, je zde i hodnota aktuálně čekajících uživatelů na odpověď testovaného serveru. Předposlední hodnota v tabulce uvádí celkovou velikost přenesených dat. A poslední hodnota vyjadřuje průměrnou hodnotu délky jedné relace na smyčku.

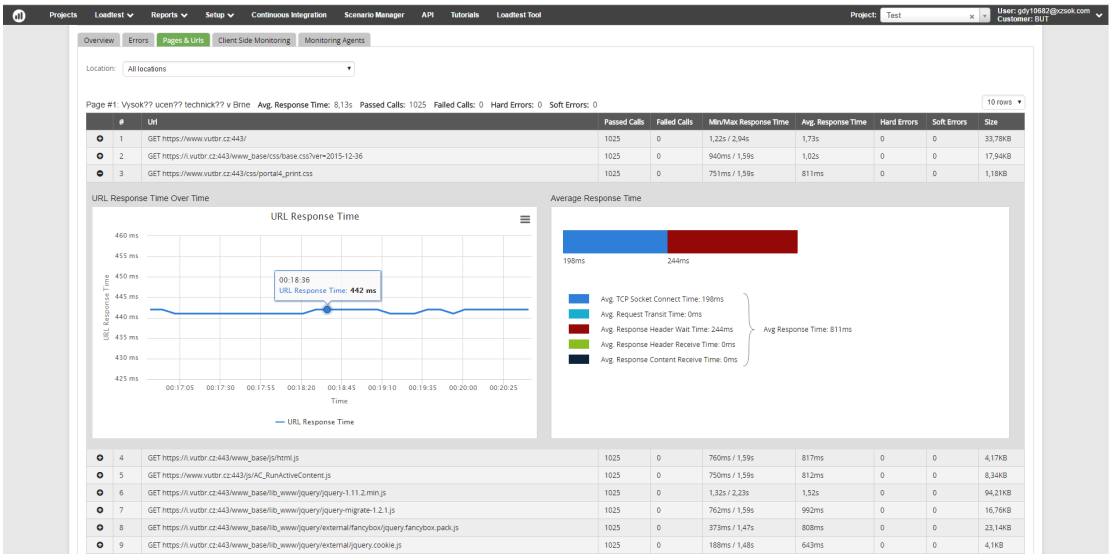
Pod tabulkou s průběžnými hodnotami jsou umístěny grafy. Všechny grafy jsou stejně jako hodnoty v tabulce průběžně doplňovány o nové hodnoty. První graf znázorňuje počet aktivních uživatelů a počet spojení čekajících na odpověď. Následuje graf s hodnotami aktuálního počtu dotazů na stránku a šířkou pásma vyjádřenou v Mbit/s. V dalším grafu je vynesen počet chyb. Tyto chyby se rozdělují na vážné a normální chyby. Vedle grafu s chybovostí se nachází graf s výčtem trvání smyčky a počtem ukončených smyček během jedné minuty. Další dva grafy zobrazují procentuální zátěž CPU a míru zaplnění RAM. Aby se tyto dva grafy vykreslovaly, je zapotřebí mít na testovaném serveru aktivního klienta Zebra Tester.

Poslední položka v sekci „Shrnutí“ je tabulka s přehledem hodnot u každé testované adresy zvlášť. V tabulce se nachází jméno testované stránky, čas věnovaný testování jedné stránky v milisekundách, průměrný čas strávený na stránce, průměrná velikost stránky, počet úspěšných dotazů na stránku, počet neúspěšných dotazů,

počet normálních chyb a počet závažných chyb.

V další sekci je vykreslen graf s počtem chyb v průběhu testování. Chyby lze filtrovat podle umístění testovacího serveru. Po kliknutí na hodnotu v grafu se zobrazí bližší specifikace chyby v tabulce pod grafem.

Ve třetí sekci jsou tabulky pro jednotlivé stránky, obsahující seznam prvků stránky, které testující server stahoval. V tabulce se nachází informace o cestě k souboru na testovaném serveru, počet úspěšných stažení, počet neúspěšných stažení, minimální, maximální a průměrný čas odezvy při požadavku na daný soubor, počet závažných chyb, počet normálních chyb a velikost stahovaného souboru. Při rozkliknutí položky (viz obr. 2.12) lze vidět graf s hodnotami o době odezvy adresy na stahovaný soubor. Vedle grafu je umístěná informační grafika, která uživatele informuje o době dílčích kroků při komunikaci s testovaným serverem a stažením požadovaného souboru.



Obr. 2.12: Bližší informace o čase stahování požadovaného souboru.

Další dvě sekce obsahují informace o monitorování ze strany klienta.

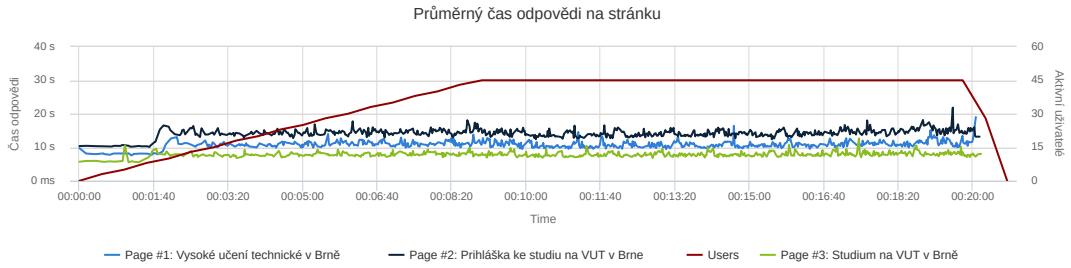
2.3.3 Konečné výsledky testu

Po ukončení testování se zpřístupní kompletní statistiky. Stránka se statistikami je opět přehledně členěna do karet (sekcí).

První sekce obsahuje souhrn. V testu byly použity dvě skupiny uživatelů a tři testované URL. Tyto aspekty dodaly výsledkům mnohem větší komplexnost. V horní části stránky se souhrnnými výsledky lze vidět počet testujících uživatelů. Jsou zde zobrazeny počty uživatelů z obou skupin. Následuje datum s časem začátku

testu i jeho ukončení. Statistiky pokračují výčtem maximálního počtu zobrazení zadané webové adresy a zadaným počtem uživatelů (oba parametry uživatel volí během nastavení testu). Další položka informuje o maximálním zobrazení stránky testujícími uživateli během jedné minuty s průměrným časem odezvy testovaného serveru. Další hodnota je počet chyb, které během testu nastaly. Poslední hodnota je maximální rychlost stahování z testovaného serveru během testu. Všechny změřené hodnoty lze porovnat mezi vytvořenými skupinami s různým počtem uživatelů.

V další části první sekce jsou zobrazeny grafy, které porovnávají nastavené skupiny uživatelů. V prvním grafu je porovnávána hodnota průměrné doby odpovědi testované stránky v milisekundách (První graf lze vidět na obr. 2.13). Následuje graf s celkovou propustností sítě v megabitech za sekundu. Třetí graf znázorňuje hodnoty stability testu – počet chyb, které během testu nastaly. Předposlední graf informuje o počtu dokončených smyček za minutu. A poslední graf znázorňuje počet zobrazení testované stránky za minutu.



Obr. 2.13: Graf s průměrným časem odpovědi na testovanou stránku.

Po grafech následuje tabulka s hodnotami statistik. Tabulka obsahuje dohromady tři sloupce (další sloupce závisí na počtu nastavených skupin uživatelů). V prvním sloupci je měřená veličina a v dalších dvou je počet uživatelů v dané skupině. V tabulce jsou hodnoty popsané v úvodní kapitole (samples, response time, latency, bandwidth a error rate). Následující seznam obsahuje hodnoty vypsané v tabulce, které nebyly v úvodu zmíněny.

Passed Loops – počet úspěšných smyček.

Failed Loops – počet neúspěšných smyček.

Total Url Errors – celkový počet dotazů na URL ukončených chybou.

Avg Session Time Per Loop – průměrná doba jednoho sezení na smyčku v milisekundách.

Avg Response Time Per Loop – průměrná doba odezvy na smyčku v milisekundách.

Avg Response Time Per Page – průměrná doba odezvy na stránku v milisekundách.

Web Transaction Rate – celkový počet úspěšných načtení webové stránky rozložený mezi všechny testující uživatele.

Total HTTP/S Calls – celkový počet HTTP/S dotazů.

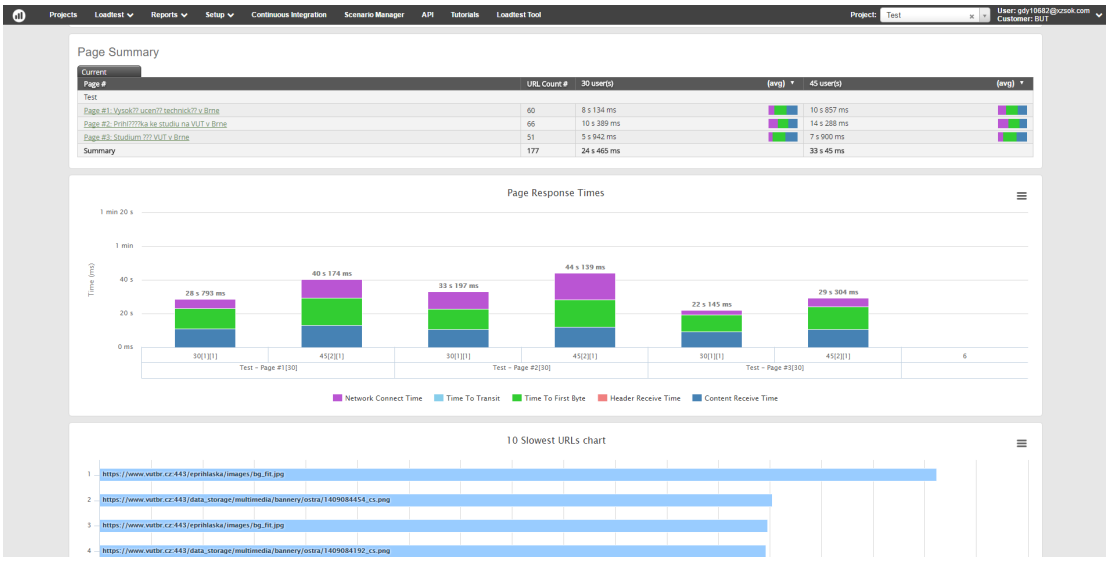
HTTP Keep-Alive Efficiency – procentuální hodnota vyjadřující počet HTTP spojení, která musela být znovu navázána.

Avg Network Connect Time – průměrná doba vytvoření nového spojení s testovaným serverem v milisekundách.

Avg Network Throughput – průměrná rychlost stahování obsahu z testovaného serveru.

Total Transmitted Bytes – celkový počet přenesených bajtů.

První sekce pokračuje tabulkou porovnávající hodnoty pro jednotlivé adresy ze zadaných a testovaných webových adres. V prvním sloupci tabulky se nachází název testované adresy, následuje sloupec s počtem URL, které jsou načítány s načítáním testované stránky (adresy obrázků, skriptů, CSS souborů, atd..). Další sloupce obsahují časové hodnoty, vyjadřující dobu trvání připojení, stažení prvního bitu, stažení hlaviček a celkový čas pro stažení všech potřebných souborů. Všechny časy jsou uváděny v milisekundách. Na tuto tabulku navazuje graf, který všechny výše uvedené hodnoty převádí do grafické podoby. Graf i s tabulkou je možné vidět na obr. 2.14.



Obr. 2.14: Graf s časy stahování částí testované stránky.

Následuje graf s vypsanými soubory, které testovacímu serveru zabraly nejvíce času na stahování. V grafu je vypsána URL k souboru a čas stahování v milisekundách. Část tohoto grafu lze vidět ve spodní části obr. 2.14.

Poslední graf zobrazuje počet chyb vyjádřených HTTP stavovým kódem. Graf je sloupcového typu a počet sloupců se odvíjí od počtu nastavených skupin testujících uživatelů.

Druhá sekce obsahuje detailní hodnoty pro každou skupinu uživatelů. Tyto hodnoty odpovídají hodnotám v tabulce a jsou totožné s hodnotami statistik z předešlé sekce. Proto nemá smysl je znovu vypisovat. Třetí sekce obsahuje seznam všech stahovaných položek se statistikami, které jsou:

Passed – celkový počet úspěšných stažení dané položky.

Failed – celkový počet neúspěšných stažení.

Avg. Time – průměrný čas stahování.

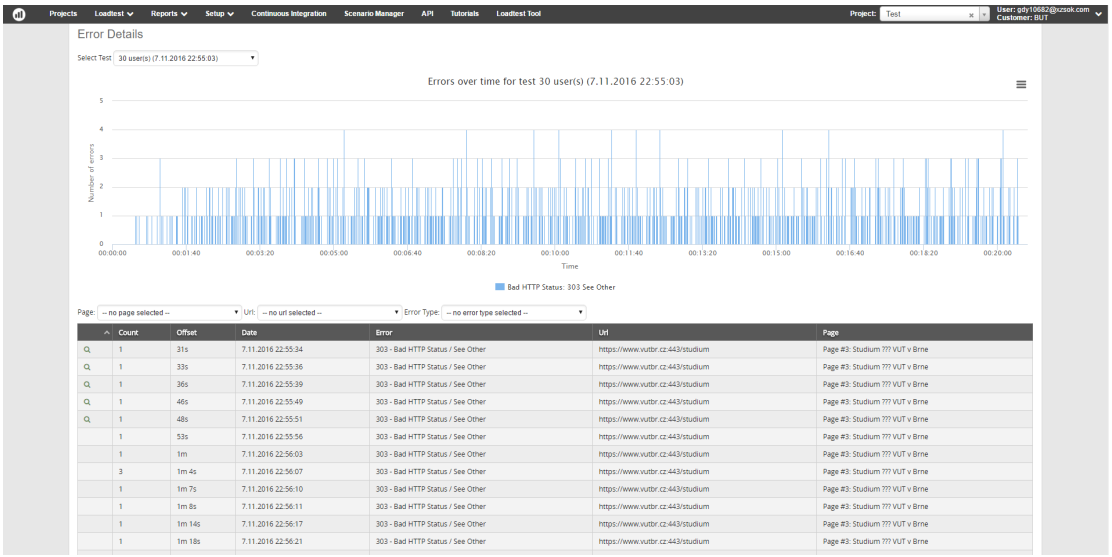
<=90 % – čas, který nastal v méně, nebo v devadesáti procentech případů.

Size – velikost v kilobajtech.

Execution Mode – způsob stažení souboru (paralelně/sekvenčně).

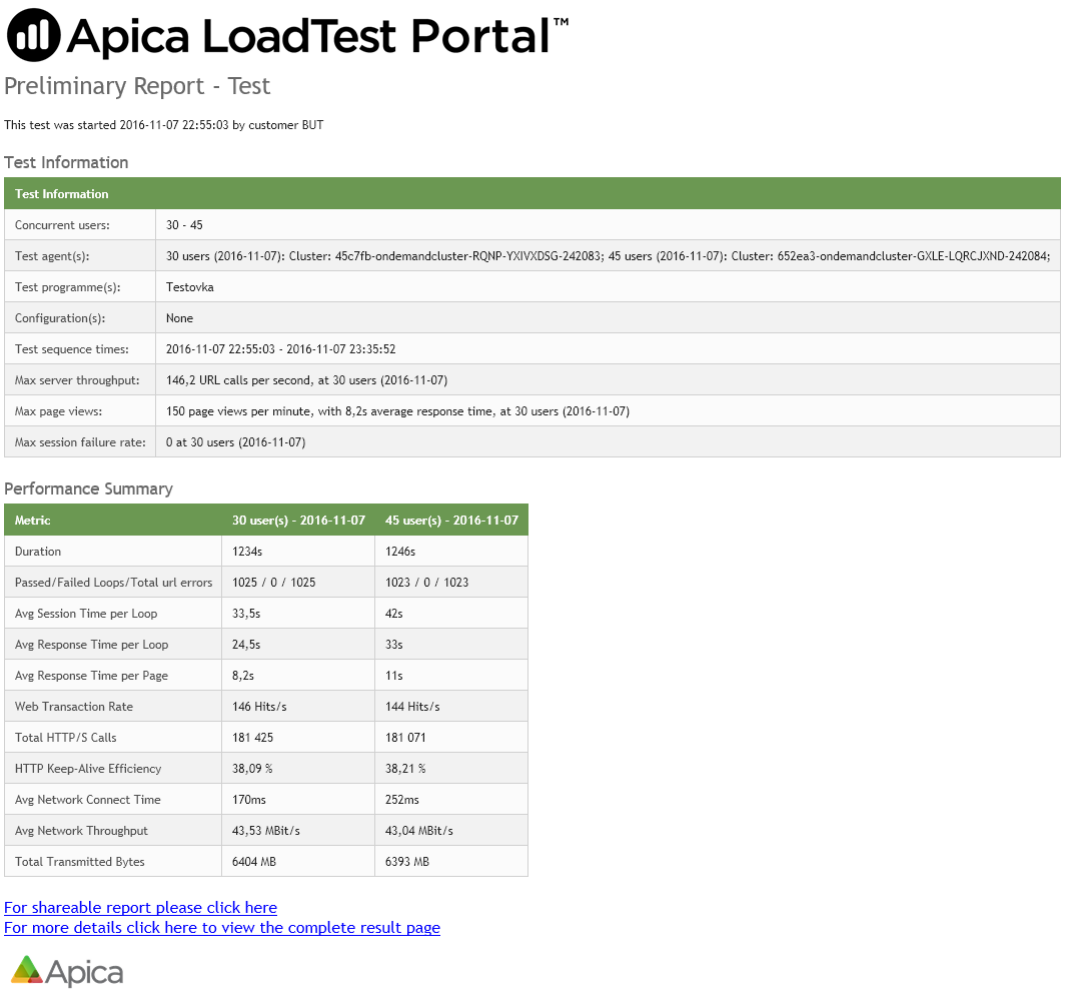
URL – unikátní adresa odkazující na stahovaný soubor.

Čtvrtá sekce obsahuje výčet chyb, které nastaly během testu. Sekce obsahuje v horní části tabulku, kde je vypsán počet chyb pro danou testovací stránku. Ve spodní části lze vybrat jednu skupinu uživatelů a po rozkliknutí se rozbalí stránka s podrobnějším výpisem chyb. Výpisu dominuje graf, který názorně ukazuje počet chyb, které nastaly během testování. Graf s počtem chyb lze vidět na obr. 2.15. Pod grafem se nachází tabulka, kde je každá chyba rozepsána dopodrobna. Tabulka obsahuje čas výskytu chyby, datum a čas jejího výskytu, typ chyby (odpovídá značení HTTP stavových kódů), odkaz na testovací stránku a název testovací stránky. Dále je zde možnost zobrazit podrobnosti chyby, kde je uveden i podrobný log chyby.



Obr. 2.15: Graf s časy při testu, kde nastala chyba.

Následují sekce, které vyžadovaly speciální nastavení nebo propojení se službami třetích stran. Jsou to sekce: AppDynamics, Transactions a New Relic. Předposlední sekcí výsledků je sekce se souhrnem nastavení. Je zde vypsáno například: jméno scénáře, počet uživatelů, délka testu, umístění serveru, atd.. Poslední sekce opět odkazuje na dodatečné nastavení, a to monitoring aktivity klienta. Toho lze docílit výše zmiňovaným softwarem „Zebra Tester“. Jako poslední je přiložena ukázka výsledků, které po ukončení testu byly odeslány na email. Email lze vidět na obr. 2.16.



Apica LoadTest Portal™

Preliminary Report - Test

This test was started 2016-11-07 22:55:03 by customer BUT

Test Information

Test Information	
Concurrent users:	30 - 45
Test agent(s):	30 users (2016-11-07): Cluster: 45c7fb-ondemandcluster-RQNP-YXIVXDSG-242083; 45 users (2016-11-07): Cluster: 652ea3-ondemandcluster-GXLE-LQRCJXND-242084;
Test programme(s):	Testovka
Configuration(s):	None
Test sequence times:	2016-11-07 22:55:03 - 2016-11-07 23:35:52
Max server throughput:	146,2 URL calls per second, at 30 users (2016-11-07)
Max page views:	150 page views per minute, with 8,2s average response time, at 30 users (2016-11-07)
Max session failure rate:	0 at 30 users (2016-11-07)

Performance Summary

Metric	30 user(s) - 2016-11-07	45 user(s) - 2016-11-07
Duration	1234s	1246s
Passed/Failed Loops/Total url errors	1025 / 0 / 1025	1023 / 0 / 1023
Avg Session Time per Loop	33,5s	42s
Avg Response Time per Loop	24,5s	33s
Avg Response Time per Page	8,2s	11s
Web Transaction Rate	146 Hits/s	144 Hits/s
Total HTTP/S Calls	181 425	181 071
HTTP Keep-Alive Efficiency	38,09 %	38,21 %
Avg Network Connect Time	170ms	252ms
Avg Network Throughput	43,53 MBit/s	43,04 MBit/s
Total Transmitted Bytes	6404 MB	6393 MB

For shareable report please click here
For more details click here to view the complete result page

Apica

Obr. 2.16: Ukázka výsledků přijatých emailem.

2.4 JMeter

Domovská stránka programu JMeter a dokumentace k programu je dostupná zde [7]. V této sekci jsou rozebrány moduly pro generování reportů v programu JMeter.

Moduly, na které tento textu odkazuje, pracují s výsledky měření. Tyto moduly jsou integrovány přímo v programu, ale je zde i možnost další moduly stáhnout. Druhou možností je naprogramovat modul vlastní.

2.4.1 View Results in Table

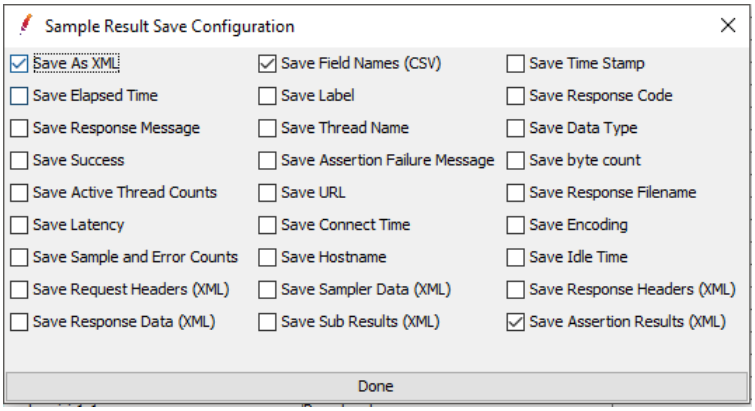
První doplněk, kterému se tato podsekce věnuje je `View Results in Table`, neboli „Zobrazení výsledků v tabulce“. Doplněk nemá žádné nastavení (kromě nastavení na obr. 2.17), avšak nabízí grafický výstup hodnot v tabulce a lze pomocí něj hodnoty ukládat do souboru, se kterým lze dále pracovat. Každá odpověď při testu je uložena na nový řádek. Způsob prezentace výsledných dat může být paměťově náročný, což je nevýhoda popisovaného doplňku. Při připojení tohoto modulu do projektu, je doporučeno jej pojmenovat. Je zde totiž možnost mít v projektu více stejných modulů a každý z nich mít nastavený jinak. Jejich pojmenování je tedy první věcí, která pomáhá moduly mezi sebou rozlišit. Dále je možné dát k doplňku komentář pro vlastní potřebu.

Důležitou součástí doplňků v programu JMeter je možnost ukládat vygenerované hodnoty do souboru. V nastavení lze vybrat XML strukturu, kterou bude následně soubor tvořen. Pokud uživatel tuto možnost nezvolí, budou data ukládána do CSV souboru. Na následujících řádcích jsou popsány možnosti nastavení formy vytvoření souboru s výslednými daty.

Save as XML – možnost uložení dat do souboru pomocí značkovacího jazyka XML.
Saved Elapsed Time – čas uplynulý od začátku testu.
Save Response Message – stavová HTTP zpráva informující o dostupnosti serveru.
Save Success – lze ukládat pouze kladně vyřízené dotazy.
Save Active Threads Counts – počet aktivních vláken (uživatelů).
Save Latency – hodnota latence, tj. prodleva odpovědi od dotazu na server.
Save Sample and Error Counts – počet vzorků a nastalých chyb.
Save Request Headers (XML) – hlavičky dotazů odesílaných na server.
Save Response Data (XML) – data obdržená od serveru.
Save Field Names (CSV) – při vytváření CSV tato volba vytvoří záhlaví s názvy hodnot.
Save Label – název, který odpovídá dotazované adrese.
Save Thread Name – název vlákna, které odeslalo dotaz na server a následně od něj dostalo odpověď.
Save Assertion Failure Message – chybové hlášení.
Save URL – webová adresa.
Save Connect Time – čas připojení.

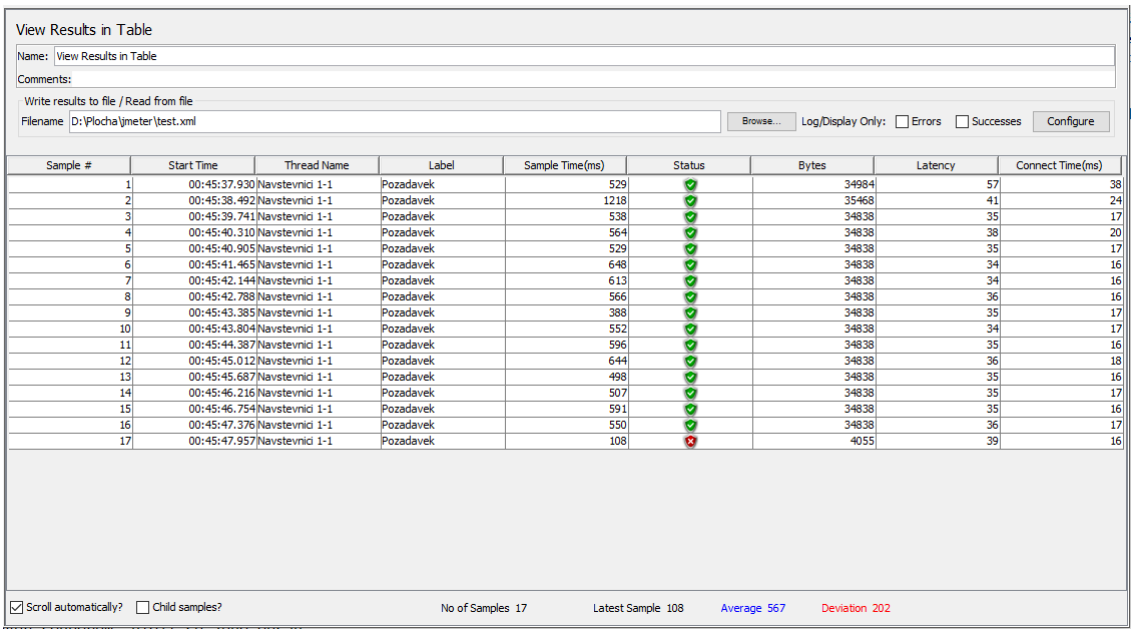
Save Hostname – název počítače, kde je program JMeter spuštěn.
Save Sampler Data (XML) – vypisuje data obsažená v souborech Cookie.
Save Sub Results (XML) – vypisuje potomky vzorků.
Save Time Stamp – časové razítko v milisekundách.
Save Response Code – HTTP stavový kód.
Save Data Type – typ obdržených dat (text, obrázek, aj.).
Save byte count – velikost odpovědi v bajtech.
Save Response Filename – odpoví-li server souborem, uloží se jeho obsah.
Save Encoding – kódování odpovědi.
Save Idle Time – doba nečinnosti.
Save Response Headers (XML) – obdržená hlavička od testovaného serveru.
Save Assertion Results (XML) – pomocí dalšího modulu lze kontrolovat stav jednotlivých dat obdržených od serveru. Pokud vznikne změna oproti předchozímu stavu, zapíše se tato změna do souboru.

Všechny výše vypsané volby lze zvolit v dialogu zobrazeném na obr. 2.17.



Obr. 2.17: Dialogové okno výběru hodnot, které bude JMeter zapisovat do souboru.

Modul disponuje možností zobrazit i potomky vzorků. To znamená, že zaznamená každé přesměrování na stránce jako jednotlivý vzorek. Zmiňovaný případ může nastat, pokud je na serveru přítomen například `htaccess` soubor. `Htaccess` soubor při dotazu na server může tento dotaz přesměrovat na jinou část stránky, než na kterou původně dotazovaná adresa směřovala. V případě potvrzené volby zobrazení potomků JMeter zaznamená dva vzorky. Vedle této volby se nachází volba automatického rolování a zobrazení sumárních hodnot v testu. Náhled modulu lze vidět na obr. 2.18.



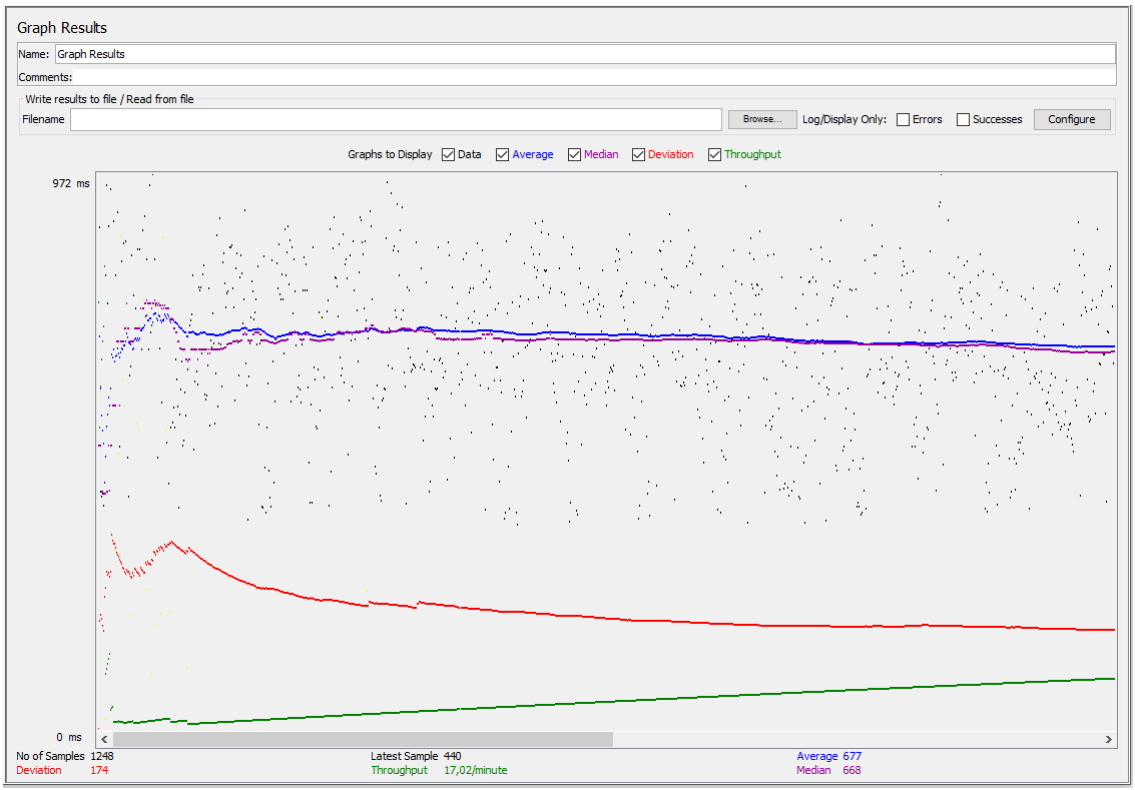
Obr. 2.18: Modul View Results in Table.

2.4.2 Simple Data Writer

Pokud uživatel nepotřebuje obrazový výstup přímo v programu JMeter, lze využít tohoto modulu. Oproti předchozímu modulu má jednu velkou výhodu – nespotřebovává velké množství paměti RAM. Popisovaný modul, stejně jako všechny moduly, které jsou v programu JMeter v základní výbavě, má totožné možnosti při zápisu do souboru. Platí tedy seznam v předešlém modulu vypsaný výše.

2.4.3 Graph Result

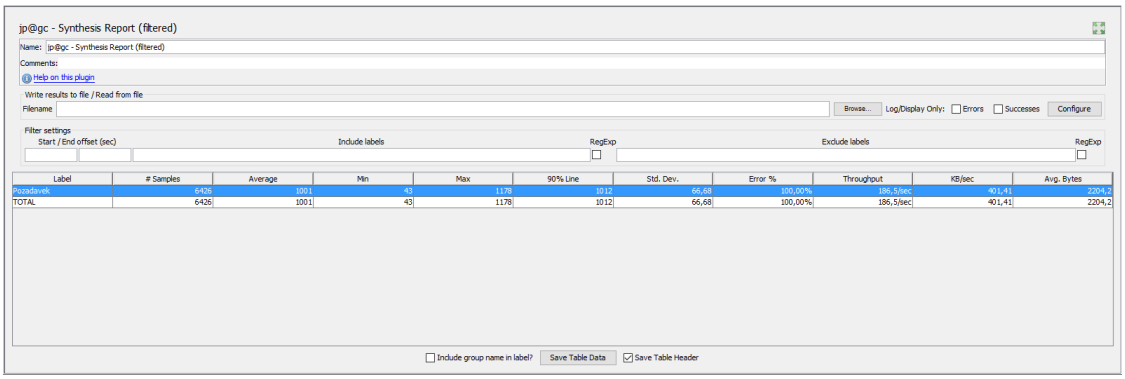
Používá-li uživatel pouze program JMeter a potřebuje výstup ve formě grafu, je zde modul `Graph Result`. Modul `Graph Result` generuje jednoduchý graf (viz obr. 2.19), který není nijak interaktivní. V tomto grafu je vyneseno dohromady pět hodnot. Hlavní hodnota má název „Data“. Hodnota „Data“ znázorňuje dobu, za kterou JMeter obdržel požadovaná data od serveru. Další hodnoty vycházejí z hodnoty „Data“. Jedná se o: průměr, medián, odchylka a poslední hodnota propustnosti.



Obr. 2.19: Modul Graph Result.

2.4.4 Plugin Manager

Protože program JMeter je vytvářen pod svobodnou licencí a je tedy jeho zdrojový kód otevřený všem zájemcům, lze pro něj vytvářet vlastní moduly. Největší sbírka modulů existuje pod záštitou společnosti Blazemeter, vyvíjející stejnojmenný nástroj popisovaný v sekci Blazemeter. Pro přístup k modulům z této databáze a jejím stažení, existuje dodatečný modul tzv. `Plugin Manager`. `Plugin Manager` usnadňuje stahování modulů a jejich instalaci tím, že oba tyto úkony obstará automaticky. V modulu si stačí požadovaný modul vybrat, kliknout na stažení a po restartu program JMeter je modul k dispozici. Pro ukázku byl pomocí `Plugin Manageru` stažen modul `Synthesis Report`. Modul `Synthesis Report` vytváří tabulku, která je obsahově podobná tabulce vyobrazené na obr. 2.3. Liší se v několika hodnotách. Jsou to minimální a maximální čas odpovědi, propustnost linky a průměrný počet bajtů. Modul `Synthesis Report` lze vidět na obr. 2.20.



Obr. 2.20: Modul Synthesis Report.

2.5 Porovnání testovaných řešení a porovnání struktur výsledků

Ze tří testovaných řešení je zřejmé, že struktury stránek s výsledky mají zažitá pravidla. Jde například o interaktivní grafy, stránku členěnou do karet a hlavní stránku obsahující souhrnné výsledky.

Všechna testovaná řešení jsou si velmi podobná. I když každé řešení je postaveno na jiné technologii, nabízí podobnou funkcionalitu a až na pár výjimek i stejné výsledky. První řešení Blazemeter nabízí logicky strukturovaný průběh testu a i jeho podání výsledků je velice přehledné. Přehlednost v podání výsledků může demonstrovat obr. 2.1. Na obrázku jsou jasně vidět hlavní hodnoty a důležité grafy, které jsou po ukončení testu žádoucí. LoadImpact se v nabízených možnostech řadí do středu mezi tři testovaná řešení. Nevýhodou řešení LoadImpact může být značně nepřehledné podání výsledků, což lze vidět na obr. 2.8. V této sekci, se menší grafy skládají pod sebe a v jednu chvíli může mít uživatel na monitoru i více než deset takových malých grafů vedle sebe. Poslední testované řešení Apica nabízí po ukončení testu nejkomplexnější výsledky. To lze vidět na obr. 2.12, kde každý řádek v tabulce po otevření obsahoval samostatný graf. V případě pokusného testu došlo k vygenerování velice komplexní zprávy s mnoha výsledky. Některé grafy se oproti zbylým dvěma řešením zdály být nepřehledné. Nepřehlednost způsobovalo mnoho vykreslených hodnot v grafu (viz obr. 2.13), které narozdíl od řešení Blazemeter nejdou skrýt. Velké plus je viděno v hlavním grafu u řešení Blazemeter (viz obr. 2.2), kde si uživatel může zobrazovat a skrývat různé potřebné hodnoty, které v dané chvíli potřebuje.

Následující Tab. 2.1 porovnává funkce jednotlivých řešení.

V Tab. 2.1 jsou vypsány vybrané možnosti a porovnání mezi testovanými ře-

	Blazemeter	LoadImpact	Apica	JMeter
Přehledné řazení výsledků	Ano	Ano	Ano	Ano
Možnost generovat PDF	Ano		Ano	
Interaktivní grafy	Ano	Ano	Ano	
Porovnání s jinými výsledky	Ano		Ano	

Tab. 2.1: Porovnání funkcí testovaných řešení.

šeními. Jak lze z tabulky vyčíst, opět jsou testovaná řešení podobná. Pouze druhé řešení Load Impact neobsahuje možnost generovat soubor s výsledky ve formátu PDF a nelze zde porovnávat s výsledky předešlých testů. Jak již bylo napsáno, všechna tři řešení jsou si velmi podobná a lze vidět mnoho podobných rysů, kterými se lze inspirovat při realizaci vlastního řešení.

3 REALIZACE VLASTNÍHO MODULU

Na základě získaných znalostí a poznatků bude v následujících odstavcích navržen vlastní modul pro program JMeter. Model bude obsahovat část pro zpracování dat, která bude řešena jako modul pro program JMeter a část prezentační jako webovou stránku, která za využití HTML, CSS a JavaScript bude vykreslovat získané hodnoty do grafů a tabulek.

3.1 Návrh a realizace modulu pro program JMeter

Program JMeter funguje z velké části na systému modulů, které lze do programu přidávat. Moduly (v adresářové struktuře projektu JMeter také nazvané jako komponenty) lze v projektu kombinovat, nastavovat a vytvářet mezi nimi spolupráci. Moduly se ve vnitřní struktuře programu JMeter rozdělují na několik skupin. Následující seznam obsahuje pouze moduly dostupné přes grafické prostředí programu JMeter (moduly určené pro práci v režimu terminálu nebo moduly určené pro jádro programu JMeter v tomto seznamu obsaženy nejsou).

- Threads (Users)
- Test Fragment
- Config element
- Timer
- Pre Procesor
- Post Procesor
- Assertion
- Listener

Protože v případě této práce se jedná o modul poskytující výsledky ve formě webové stránky, bylo zvoleno vytvoření modulu pro kategorii Listeners. Tato kategorie zastřešuje moduly pro práci s výsledky testů. Jsou zde obsaženy moduly, které dokáží exportovat výsledky do souboru s formátem XML, nebo CSV. Dále obsahuje moduly pro okamžité vykreslování grafů, tabulek s průběžnými výsledky atd. Při bližším seznámení bylo zjištěno, že program JMeter obsahuje funkci, která dokáže na základě výsledků uložených v souboru CSV vygenerovat webovou stránku. Tato funkcionalita se jmenuje Dashboard a je dostupná pouze z rozhraní terminál programu JMeter. Této funkcionality bylo využito při tvorbě vlastního modulu.

Popis fungování funkce Dashboard

V předchozím odstavci bylo nastíněno fungování funkce Dashboard, dále tedy bude funkce popsána více. Tuto funkci lze vyvolat pouze skrze terminál nebo příkazový řádek (podle toho v jakém systému se uživatel nachází). Funkci lze spustit buďto bezprostředně po testu, nebo již z hotového souboru ve formátu CSV, který obsahuje výsledky testu. Spuštění funkce Dashboard bezprostředně po skončení testu lze vyvolat zadání příkazu:

```
jmeter -n -t <a> -l <b> -e -o <c>
```

kde parametr „a“ je konfigurační soubor testu programu JMeter. „b“ je již vzpomínaný CSV soubor obsahující výsledky testu a „c“ je cesta ke složce pro umístění výsledné webové stránky. Tento příkaz test spustí a následně po něm je vygenerována webová stránka. Druhý příkaz vytvoří webovou stránku s výsledky z již hotového CSV souboru.

```
jmeter -g <b> -o <c>
```

Zde se zadávají stejné parametry jako u předchozího příkazu. Funkce Dashboard generuje stránky z již předpřipravených šablon. Tyto šablony se nachází ve složce */bin/report-template*. Šablona má klasickou strukturu HTML stránky s rozšířenou funkcionalitou pomocí jazyka JavaScript. Pokud program JMeter zaznamená požadavek o vygenerování reportu do webové stránky, funkce Dashboard začne automaticky v této složce procházet všechny soubory. Při procházení se zaměřuje na soubory s koncovkou „.fmkr“. Pokud je takový soubor nalezen, je otevřen a funkce Dashboard projde jeho obsah. Je-li soubor s takovou koncovkou obsažen v šabloně, lze do tohoto souboru začlenit předem definované příkazy pro funkci Dashboard. Tyto příkazy odkazují na data, která jsou uložena ve formátu JSON (JavaScript Object Notation). Dashboard při vytváření webové stránky tento příkaz smaže a na jeho původním místě zanechá požadovaná data, se kterými lze následně na stránce pomocí jazyka JavaScript dále pracovat. Protože funkce Dashboard bohužel neobsahuje všechna potřebná data, která se běžně v reportech využívají (viz Parametry zátěžového testování), bylo nutné další potřebné hodnoty implementovat v modulu dodatečně.

Reportovaná data

V následujícím seznamu budou uvedena data, která nejsou obsažena ve funkci Dashboard:

- `Request Label` – jméno požadavku HTTP.
- `\#Users` – počet uživatelů na jeden požadavek HTTP.

- `\#Samples` – počet odeslaných požadavků.
- `Average` – průměrná hodnota.
- `Min` – minimální hodnota.
- `Max` – maximální hodnota.
- `Std. Dev.` – směrodatná odchylka.
- `Error %` – procentuální vyjádření počtu chyb.
- `Throughput` – propustnost.
- `Recieved KB/s` – počet přijatých KB za jednu sekundu.
- `Sent KB/s` – počet odeslaných KB za jednu sekundu.
- `Average bytes/s` – průměrná hodnota.
- `Latency Avg.` – průměrná hodnota latence.
- `Connect time` – průměrná hodnota času připojení při spojení.
- `Avg. Response Time (ms)` – průměrná doba času odpovědi.
- `URLs` – webové adresy stránek, se kterými jeden požadavek HTTP komunikoval.

Pro všechna výše vypsaná data lze využít třídu Calculator. Tato třída poskytuje průběžná data, která jsou dostupná pro každý požadavek HTTP zvlášť. Dále lze také získat statistiky pro všechny HTTP požadavky dohromady. Aby třída korektně fungovala, potřebuje předešlá data k výpočtu dat aktuálních. Pokud jí tato data nejsou poskytnuta, jsou pouze hodnoty `Avarege`, `Min`, `Max` a `Std. Dev`. správně. Ostatní hodnoty jsou počítány z předchozích dat. Druhou třídou, poskytující data výsledků měření, je třída SamplingStatCalculator, která je využita k výpočtu percentilů odezvy testovaného serveru v 90 %, 95 % a 99 % případů. Tato třída funguje stejně jako třída Calculator. Celkovému popisu práce s oběma třídami se bude věnovat kapitola popisu práce s daty.

3.1.1 Základní stavba modulu

Ve výpisu 3.4 lze vidět základní konstrukci nově vytvořeného modulu. Samotný kód v konstrukci zatím nic neumí, respektive umí zobrazit předdefinovaný panel tzv. TitlePanel, který slouží k definování cesty k souboru pro uložení výsledků z probíhajícího testu. Lze v něm dále zvolit parametry, které se budou do souboru ukládat. Konfigurační okno parametrů je možno vidět na obr. 2.17. Tento panel obsahuje většina modulů v programu JMeter, není však bezprostředně nutné jej zobrazit.

Výpis 3.1: Základní konstrukce modulu WebGenerator.

```
1  package org.apache.jmeter.visualizers;
2  import java.awt.BorderLayout;
3  import org.apache.jmeter.samplers.SampleResult;
4  import
       org.apache.jmeter.visualizers.gui.AbstractVisualizer;
5
6  public class WebGenerator extends AbstractVisualizer {
7      private static final long serialVersionUID = 240L;
8
9      public WebGenerator() {
10         init();
11         setName(getStaticLabel());
12     }
13
14     @Override
15     public String getLabelResource() {
16         return "web_generator_title"; // $NON-NLS-1$
17     }
18
19     private void init() {
20         setLayout(new BorderLayout());
21         setBorder(makeBorder());
22         add(makeTitlePanel(), BorderLayout.SOUTH);
23     }
24
25     @Override
26     public void clearData() {
27         // NOOP
28     }
29
30     @Override
31     public void add(SampleResult sample) {
32         // NOOP
33     }
34 }
```

Nyní lze přistoupit k popisu samotného kódu. Na prvních čtyřech řádcích kódu lze vidět vložení (import) externích tříd. Protože vytvářený modul spadá do skupiny Visualizers, je definován balíček visualizers. Dál jsou importovány dvě třídy BorderLayout a SampleResult. První třída řeší rozvržení prvků při vykreslení modulového

GUI. Druhá třída umožňuje čtení hodnot jednotlivých vzorků. Práce s jednotlivými vzorky bude popsána později. Poslední importovaná třída umožňuje zobrazení již zmiňovaného „TitlePanel“.

Po importu všech tříd následuje definování samotné třídy modulu. Zde je již v základu rozšiřující třída AbstractVisualizer. Na řádku číslo sedm je definováno sériové číslo modulu. Toto číslo definuje prvky, které lze vykreslit a přiřadit jim funkčnost mimo funkčnost modulu, ale lze jimi ovládat přímo metody v rozšiřující třídě. V případě tohoto modulu třída AbstractVisualizer. V případě modulu WebGenerator se však funkčnost vykreslování těchto prvků využívat nebude, avšak toto UID je použito. Následuje konstruktor WebGenerator, který po spuštění modulu obstará základní inicializaci. To je také první metoda, kterou konstruktor volá. Je pojmenována `init()` a obstarává vykreslení TitlePanelu. Ten se vykreslí do nastaveného uspořádání (Layout), ve kterém je definovaný rám (Border), kde je umístěn TitlePanel do vrchní oblasti modulu. Toto umístění je dáno zaběhlým standardem, který dodržují všechny moduly, jež obsahují TitlePanel. Po provedení metody `init()` následuje nastavení jména modulu. V případě tohoto modulu se jedná o jméno „Web Generator“, které bylo zvoleno na základě jeho primárního funkčního zaměření. Tato metoda zavolá metodu getStaticLabel, ve třídě AbstractVisualizer, která zavolá metodu `getLabelResource()`. Tato metoda má v parametru udaný řetězec, který je uložen spolu se jménem modulu v souboru „messagess.properties“. Tento soubor má několik jazykových mutací a slouží k uložení textových řetězců s názvy tlačítek, voleb v menu, informačních hlášek a prakticky celého programu JMeter. Pomocí tohoto souboru lze vytvořit novou jazykovou mutaci programu JMeter, aniž by musel překladatel zasáhnout do programového kódu.

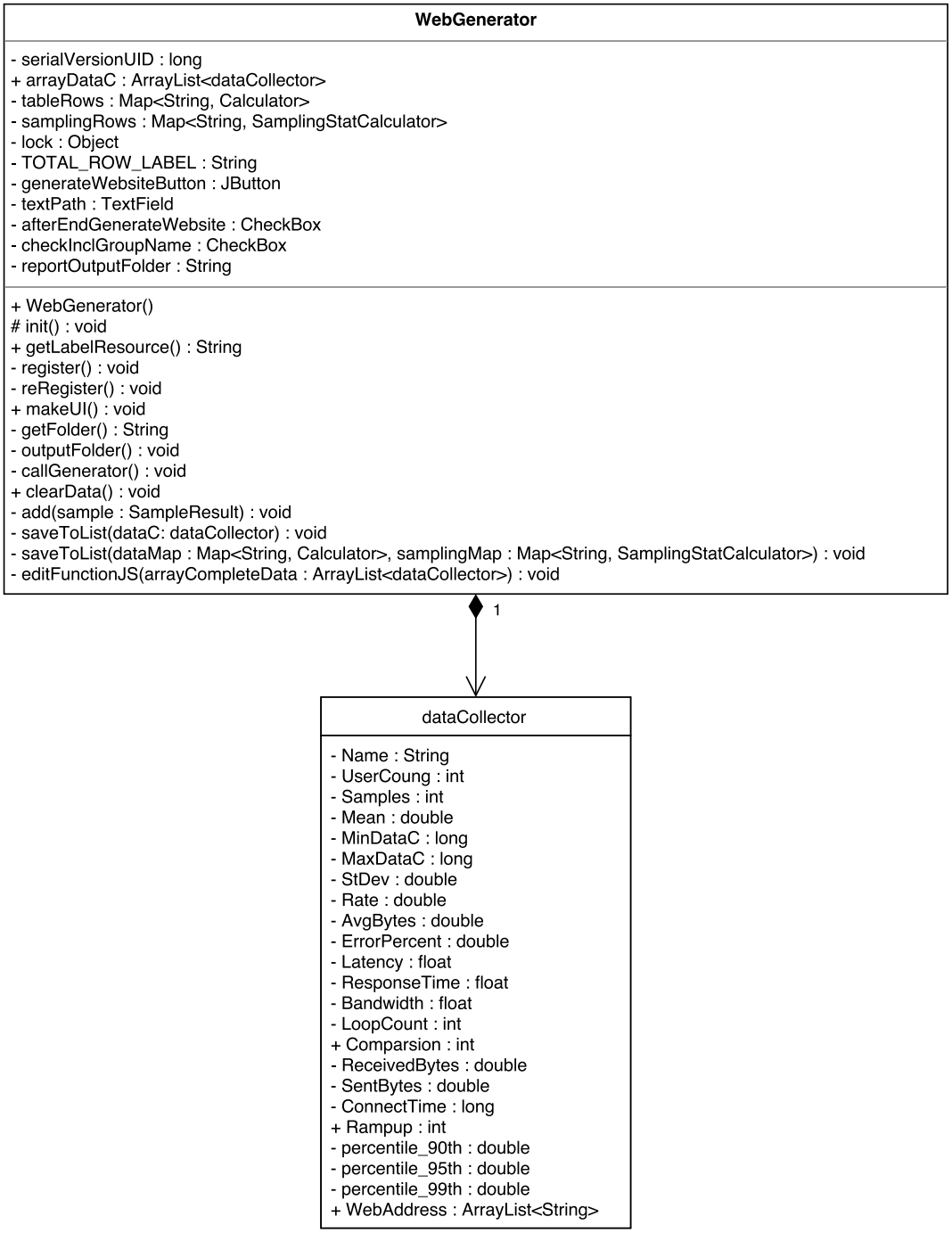
V třídě se také nacházejí další dvě metody `clearData()` a `add()` s parametrem SampleResult. Tyto metody budou popsány v další části, kde bude podrobněji vysvětleno jejich využití.

3.1.2 Struktura vlastního objektu

Na základě zjištěných údajů v první části věnující se zátěžovému testování (viz kapitola Parametry zátěžového testování) byl navržen objekt, který obsahuje všechny potřebné proměnné. Tento objekt byl pojmenován „dataCollector“. Strukturu lze vidět na obr. 3.1. Objekt obsahuje ke každé proměnné getter a setter. Více o metodách getter a setter: [4][12].

3.1.3 Odchycení události

Pro zjištění startu a konce testu je využita třída TestStateListener. Tato třída při přetížení ve třídě WebGenerator dokáže při spuštění testu a při jeho konci vyvolat požadovanou metodu. Třída TestStateListener funguje pouze po jejím zaregistro-



Obr. 3.1: UML diagram třídy.

vání. Toto zaregistrování je potřeba provést po každém zapnutí a vypnutí testu. Jak je možné vidět ve výpisu číslo 3.2, je TestStateListener zaregistrován přímo v jádru programu JMeter, který tento objekt typu `listener` obsluhuje. Při testování se však přišlo na to, že je potřeba zaregistrovat celou metodu `register()`, která obsahuje submetody `testEnded()` a `testStarted()`. Pokud byla provedena registrace pouze při incializaci modulu, fungovalo by odchycení události pouze jednou. Proto je nutné provést registraci pokaždé, kdy se test ukončí. Opětovná registrace se provádí v metodě `testEnded()` a je prováděna rekurzivně.

Výpis 3.2: Zaregistrování čekání na událost v *TestStateListener*.

```
1 StandardJMeterEngine.register(listen);
```

3.1.4 Práce s daty

Data jsou shromažďována pro každý požadavek HTTP zvlášť. Ve třídě je umístěn `ArrayList`, ve kterém jednotlivá položka (řádek) odpovídá jednomu požadavku HTTP. `ArrayList` je typu dataCollector. Aby bylo možné shromažďovat data obsažená v objektu dataCollector, je potřeba využít zmiňované třídy Callculator a SamplingStatCalculator. Samotná práce s daty začíná v metodě `add()`.

Tato metoda je volána přímo jádrem programu JMeter. V parametru se nachází třída SampleResult. Tato metoda je volána pokaždé, dorazí-li odpověď od testovaného serveru. V třídě SampleResult jsou uloženy informace o výsledku odpovědi. Hlavní informace uložená v této třídě je název požadavku HTTP, který je dále předáván třídám Callculator a SamplingStatCalculator.

Kód v ukázce 3.3 je obsažen v metodě `add()`. Na prvním řádku z ukázky 3.3 se ukládá do pomocné proměnné typu String název požadavku HTTP. Následuje inicializace tříd Callculator a SamplingStatCalculator. Pro korektní chod programu je potřeba synchronizovat třídy. Tento úkon provede řádek číslo čtyři. Na dalších dvou řádcích se z kolekce typu Map vyberou podle klíče sampleLabel (název požadavku HTTP) objekty Callculator a SamplingStatCalculator. Objekt Callculator je uložen v kolekci tableRows a SamplingStatCalculator se nachází v kolekci samplingRows.

Výpis 3.3: Práce s třídami *Callculator* a *SamplingStatCalculator*.

```
1  final String sampleLabel = sample.getSampleLabel(false);
2  Calculator calc;
3  SamplingStatCalculator samplingCalc;
4  synchronized (lock) {
5      calc = tableRows.get(sampleLabel);
6      samplingCalc = samplingRows.get(sampleLabel);
7      if (calc == null) {
8          calc = new Calculator(sampleLabel);
9          tableRows.put(calc.getLabel(), calc);
10     }
11     if (samplingCalc == null) {
12         samplingCalc = new
               SamplingStatCalculator(sampleLabel);
13         samplingRows.put(samplingCalc.getLabel(),
               samplingCalc);
14     }
15 }
16 synchronized (calc) {
17     calc.addSample(sample);
18     samplingCalc.addSample(sample);
19 }
20 Calculator tot = tableRows.get(TOTAL_ROW_LABEL);
21 SamplingStatCalculator samplingtot =
      samplingRows.get(TOTAL_ROW_LABEL);
22 synchronized (tot) {
23     tot.addSample(sample);
24     samplingtot.addSample(sample);
25 }
```

V sekci Reportovaná data bylo uvedeno, že třídy Callculator a SamplingStatCalculator potřebují ke své práci předešlá data. Na řádcích pět a šest se tato data předají a kód pokračuje kontrolou obsahu objektů. Pokud jsou objekty prázdné (žádná data nebyla předána) znamená to, že test je na začátku a žádné výsledky ještě neobdržel. Pokud je tedy podmínka o prázdnosti objektů platná, v mapách se vytvoří nové řádky korespondující s názvem požadavku HTTP. Tyto řádky jsou posléze naplněny. Posledním krokem v ukázce číslo 3.3 je sumarizace všech požadavků HTTP do jednoho řádku s názvem „TOTAL“, který obsahuje kompletní statistiky za celý test. Tyto statistiky jsou průměrované.

Calculator a SamplingStatCalculator však nedokáží vygenerovat všechny potřebné statistiky, a proto se v metodě `add()` pokračuje plněním vlastního objektu.

V tomto kroku je využíván parametr typu SampleResult, metody `add()`. Z tohoto parametru lze zjistit hodnoty `Latency`, `Response Time`, `Connect Time`, `Group Threads` a `URL`. Tyto hodnoty jsou následně uloženy do nově vytvořeného objektu typu dataCollector. Po uložení hodnot následuje druhé uložení pro průměrné statistiky pro objekt s názvem „TOTAL“.

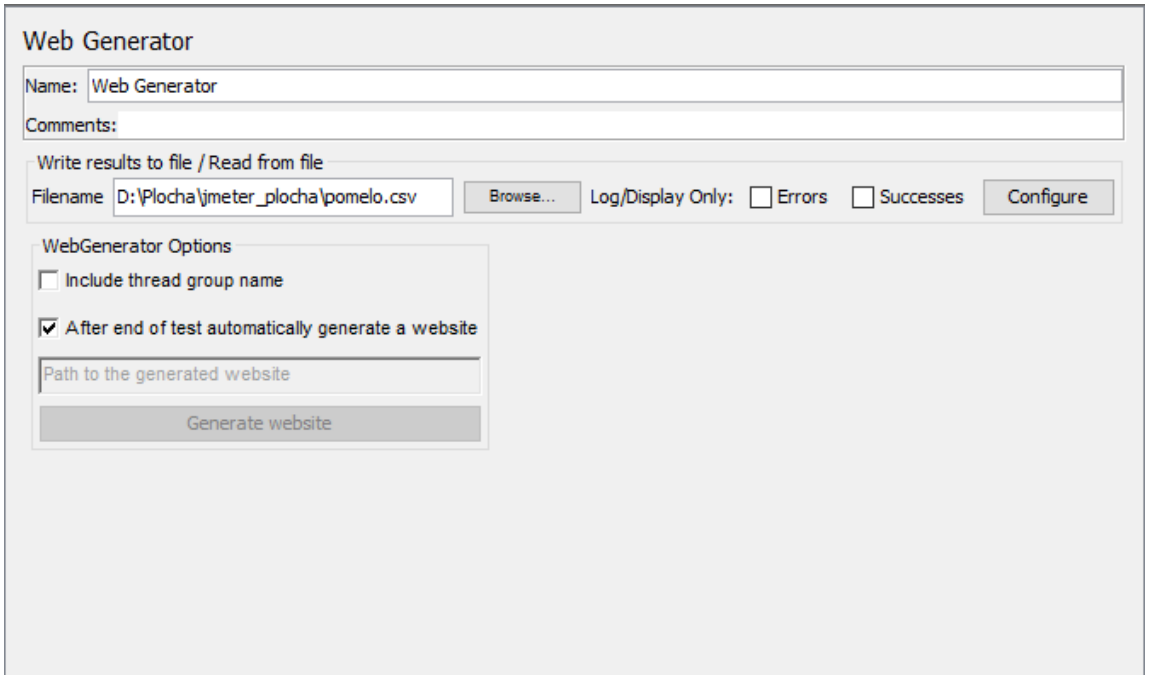
Celé volání metody `add()` je pouze pro jeden požadavek HTTP a je tedy potřeba získané statistiky uložit. K tomuto účelu je vytvořen `ArrayList` s názvem arrayDataC. Na konci metody `add()` se zavolá metoda `saveToList()` s parametrem dataCollector. V prvním kroku metoda zjistí, zdali není `ArrayList` prázdný. Pokud ano, na první pozici uloží předaný objekt. Není-li `ArrayList` prázdný, je nalezen řádek podle jména požadavku HTTP a na jeho místo se uloží aktuální hodnoty.

Takto probíhá funkce modulu po dobu testování. Následující odstavce se budou věnovat akcím po ukončení testu.

Pokud nastane konec testu, je jádrem programu JMeter zavolána metoda `testEnded()`. Prvním krokem po ukončení testu je zavolání metody `callGenerator()`. V této metodě je volána další metoda s názvem `outputFolder()`. Metoda `outputFolder()` zapíše cestu ke složce, kde je uložen soubor s výsledky testu ve formátu CSV do proměnné reportOutputFolder. K této cestě je připojen identifikátor ve formě aktuálního data s časem, ve kterém se test ukončil. Tento identifikátor je ve formátu „RRRRMMDDhhmm“, kde R jsou číslice roku, M – měsíce, D – dne, h – hodiny a m – minuty. V kapitole Popis fungování funkce Dashboard, byla popisována funkce Dashboard, která je využita pro vygenerování výsledného reportu ve formě webové stránky. Po uložení cesty k cílové složce do globální proměnné reportOutputFolder se vytvoří proměnná typu „File“. Následně se ověří, že cesta je správná. Po ověření se nastaví globální proměnná „JMETER_REPORT_OUTPUT_DIR_PROPERTY“ s dříve vytvořenou cestou k cílové složce. Je-li vše nastaveno, začne fáze generování reportu ve formě webové stránky. I v případě volání této třídy přímo z modulu fungují vnitřní procesy generování stránek stejně jako při volání z konzole viz Popis fungování funkce Dashboard. Ještě před zavoláním třídy `ReportGenerator` je nutné ji inicializovat. Tento proces zahrnuje předání cesty k cílové složce. Po předání cesty zbývá pouze spuštění generátoru. Po ukončení generování je v grafickém prostředí, v prvku TextField se jménem „textPath“ zobrazena cesta k cílové složce a pokud je zaškrtnut checkBox se jménem „afterEndGenerateWebsite“, je automaticky otevřena složka s vygenerovanou stránkou.

Vygenerovaná webová stránka však neobsahuje data, která byla zaznamenávána v průběhu testu popisovaným modulem. Po vygenerování stránky se tedy zavolá metoda `editFunctionsJS()` s parametrem `ArrayList<dataCollector>` s názvem arrayCompleteData. V tomto stavu nejsou statistiky v ucelené formě, a proto je po-

třeba je všechny sloučit na jedno místo. K tomuto účelu slouží metoda `saveToList()` s parametry Map<String, Calculator> a Map<String, SamplingStatCalculator>. Jakmile se tato metoda zavolá, projde obě předané kolekce a všechna data jsou předána do `ArrayListu` se jménem arrayDataC, kde opět každý řádek reprezentuje jeden požadavek HTTP. Pro souhrnné statistiky je vytvořen objekt s názvem „TOTAL“. Objekt „TOTAL“ se v `ArrayListu` nenachází, ale je samostatně. Po tomto předání dat se pokračuje v metodě `editFunctionsJS()`. Tato metoda má za úkol zapsat všechna data, se kterými doteď modul pracoval do dříve vytvořeného souboru „functions.js“ ve složce s vygenerovanou webovou stránkou. Pomocí třídy BufferedWriter se otevře soubor „functions.js“ a postupně do něj zapíše všechna data. Tato data jsou evidována do pole polí, kde každé vnořené pole reprezentuje jeden požadavek HTTP, plus pole se souhrnnými statistikami. Zápis probíhá procházením `ArrayListu` arrayCompleteData a postupným zapisováním hodnot do souboru. Následuje zápis souhrnných statistik z objektu total. Nyní je web vygenerovaný a s kompletními daty. Na obr. 3.2 lze vidět výsledné uživatelské prostředí modulu.



Obr. 3.2: GUI modulu WebGenerator.

3.2 Realizace šablony webových stránek

Hlavním úkolem práce je vytvoření šablony pro report ve formě webových stránek. Stránky jsou vytvořeny pomocí technologií HTML, CSS a JavaScript. Popis vytvá-

ření těchto stránek je rozdělen do dvou sekcí. První sekce bude popisovat vizuální část, tvořenou pomocí HTML a CSS. Druhá část bude obsahovat část, která se postará o prezentaci předaných dat z modulu a funkce Dashboard. Tato část bude tvořena pomocí technologie JavaScript.

3.2.1 Návrh vzhledu webové stránky

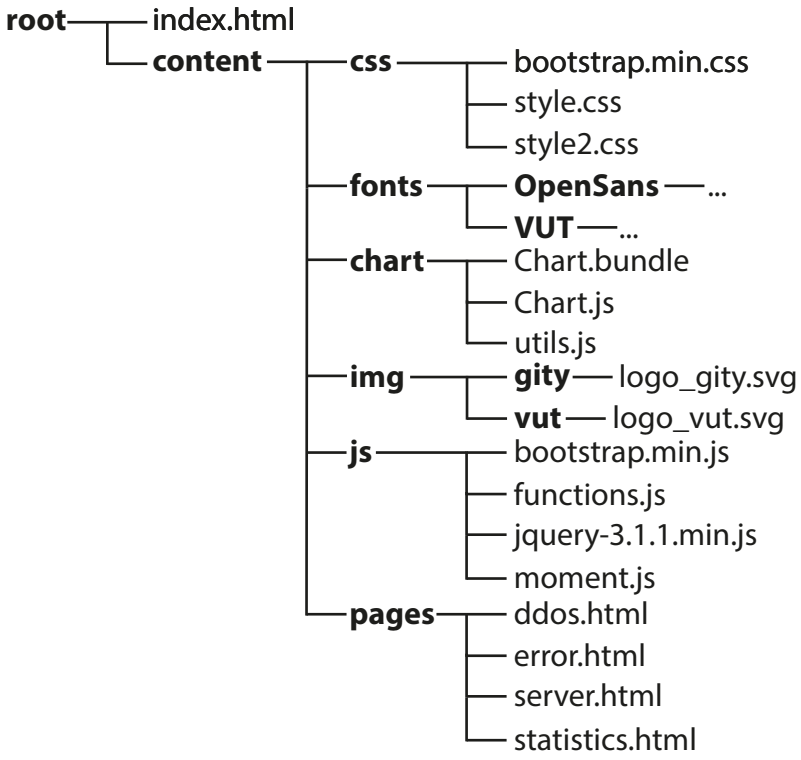
Po předešlé analýze řešení Blazemeter, LoadImpact a Apica bylo rozhodnuto, že hlavní strana bude obsahovat souhrnné informace o testu. Tyto informace budou v průměrných a maximálních hodnotách. Dále se na hlavní stránce bude nacházet graf s nejdůležitějšími hodnotami. Toto menu bude odkazovat do podobných sekcí, jako byly vidět u dříve analyzovaných řešení. Pod menu se v levé části bude nacházet informační panel s důležitými hodnotami z testu. Tyto hodnoty budou průměrného nebo maximálního charakteru. Po pravé straně od informačního panelu bude zobrazen graf s některými hodnotami z informačního panelu.

Ve vektorovém programu Illustrator byl proveden základní návrh vzhledu stránky s výsledky. Jak je vidět na obr 3.3, většinu prostoru na stránce zabírá graf, kde bude zobrazena většina důležitých hodnot.

Obr. 3.3: Návrh webu.

Adresářová struktura reportu

Po ukončení testu se vygeneruje webová stránka s následující adresářovou strukturou (kde normální písmo značí soubor a tučné značí adresář):



Obr. 3.4: Stromová struktura adresáře s vygenerovanou webovou stránkou.

3.2.2 Vizuální část

Vizuální části webové stránky se rozumí ta část, která je vytvořena pomocí jazyků HTML a CSS. Soubory napsané v těchto dvou jazycích definují výsledný vzhled stránky. Jako pomocný aplikační rámec (framework) byl zvolen Bootstrap, který popsán v následující podsekci a pro vykreslování grafů byl využit aplikační rámec ChartJS.

Bootstrap

Při návrhu stránek lze využít mnoho technologií, které jsou dostupné na Internetu. Tyto technologie velmi ulehčují následnou tvorbu. V případě tohoto návrhu byl zvolen CSS aplikační rámec (framework) Bootstrap. Tento CSS aplikační rámec poskytuje hotové styly, které dokáží vytvořit moderní a ucelený vzhled webu. Dále řeší responzivnost stránek a grafickou jednotnost.

ChartJS

Druhý využitý framework je určen k vykreslování grafů. Jeho název je ChartJS. Zvolen byl pro jeho způsob licencování, které je typu „MIT License“. Tento framework má stejně jako předchozí Bootstrap otevřený kód (opensource). Druhý důvod pro zvolení tohoto aplikačního rámce byl jeho design, který je jednoduchý a moderní. Tento framework je naprogramován pomocí technologie HTML, CSS a JavaScript, což odpovídá programovacím jazykům použitým v této práci. Práce s tímto aplikačním rámcem bude popsána v dalších odstavcích.

3.2.3 Popis vizuální části

Ze stromové struktury v obr. 3.4 lze vyčíst, že webová stránka obsahuje dohromady pět souborů ve formátu HTML. Hlavní soubor byl dle obecných konvencí pojmenován jako „index.html“. Tento soubor je míněn jako hlavní soubor celé stránky. Všechny HTML soubory jsou programovány ve stejnojmenném jazyce. Základní kostra je pro všechny soubory stejná. Základní kostrou se rozumí:

- Nastavení jazykové mutace na český jazyk. Kódování stránky ve formátu UTF-8.
- Importování CSS souborů. CSS soubory jsou importovány, jak vlastní (style.css – světlý vzhled, style2.css – tmavý vzhled), tak soubory aplikačního rámce Bootstrap.
- Další importované soubory do stránky jsou JavaScriptové knihovny.
- Opět vlastní a knihovny třetích stran.
- Knihovny třetích stran zahrnují aplikační rámce Jquery verze 3.1.1. (vyžadováno aplikačním rámceme Bootstrap), Bootstrap, ChartJS a MomentJS.
- Dále knihovna MomentJS dokáže pracovat s mnoha různými formáty času. Tato knihovna je použita pro práci s časovým razítkem při vkládání dat do grafů. Více o možnostech teto knihovny zde: [10].
- Dále se v kostře nacházejí podmínky pro částečnou kompatibilitu se staršími prohlížeči typu Internet Explorer od verze 9 a nižší. Zde je kompatibilita pouze v rámci aplikačního rámce Bootstrap.

Po importu všech potřebných knihoven a souborů s kaskádovými styly následuje tělo stránky. První je programována nabídka menu. Tato nabídka je tvořena HTML elementem „nav“, kde je definováno pomocí atributu typu „class“ s hodnotou „navbar-inverse“ jeho tmavě šedá barva. Dále jsou obsaženy vnořené elementy typu „div“. Jako další je vnořen element typu „ul“, který definuje seznam. Každá položka v seznamu je element typu „li“. Jednotlivé položky v tomto seznamu definují položky ve vykreslené nabídce menu na webové stránce. V seznamu se nachází dohromady pět položek pro pět stránek (pět různých HTML souborů). Aplikační

rámec Bootstrap umožňuje zvýraznění aktuálně otevřené stránky. Toto zvýraznění lze vyvolat pomocí přiřazením atributu typu „class“ s hodnotou „active“. Každá položka seznamu obsahuje vnořený element „a“ s atributem „href“, kde jeho hodnota je cesta k příslušnému souboru odkazované stránky. Další nabídka menu se nachází na pravé straně. Zde je k výběru změna stylu stránky na tmavou, nebo světlou. Tato nabídka je vytvořena stejně jako předchozí. Pouze se liší element „a“, kde je navíc atribut „onclick“, který odkazuje na funkci v JavaScriptové knihovně. Poslední dvě položky jsou loga VUT a GiTy.

Tímto končí společná kódová část (kostra), která je pro všechny stránky stejná. V dalších podsekcích bude popsán kód pro jednotlivé stránky.

Hlavní stránka – index.html

V horní části stránky je název testu, který je převzat z názvu CSV souboru s výsledky testu. Dále na hlavní stránce jsou tři hlavní prvky. První je sloupec se souhrnnými hodnotami. Tento sloupec je tvořen z několika do sebe vnořených elementů typu „div“. Hlavní element má atribut „class“ s hodnotou odkazující na rozvržení stránky a jejímu následnému chování při změně proporcí zobrazovací plochy (změna velikosti okna prohlížeče). Tyto hodnoty atributů jsou blíže specifikovány v dokumentaci aplikačního rámce Bootstrap na jeho oficiálních stránkách [2]. V levém sloupci hodnot, které byly vybrány na základě sekce Hlavní parametry ve výsledcích zátěžového testování. Každá položka je definována pomocí elementu typu „div“, který má atribut s hodnotou, která odpovídá typu vykresleného parametru výsledku. Element div má v sobě vnořené tři elementy typu „span“, kde první element obsahuje hodnotu výsledku. Zde je také atribut s názvem „id“, který je použit v JavaScriptové knihovně k předání hodnoty. Druhý element obsahuje veličinu hodnoty (například: „users“, „errors“, „%“, atd. . .). Poslední element obsahuje slovní popis vypsané hodnoty.

Po uzavření všech elementů náležících k levému panelu, následuje jeden element typu „div“, který je společný pro zbylé hlavní prvky. Společné zapouzdření do jednoho divu bylo zvoleno z důvodu stejné šířky prvků. Tento element opět obsahuje atribut vycházející ze standardu aplikačního rámce Bootstrap. Druhý hlavní prvek na stránce je informační panel. Tento panel se nachází v horní části stránky. Jsou v něm obsaženy informace o adrese testované stránky, délce testu, začátku testu a jeho konci. Kódové zobrazení panelu je řešeno stejně jako panel s hlavními výsledky testu na levé straně. Pro každou hodnotu je tedy vytvořen element typu „div“, který obsahuje elementy typu „span“. Všechny hodnoty v tomto panelu jsou vykreslovány pomocí JavaScriptové knihovny.

Poslední hlavní prvek na stránce je panel s grafy. Tento panel obsahuje dva vnořené elementy typu „div“. První vnořený element obsahuje definici záložek pou-

žitých v tomto panelu. Záložky jsou, stejně jako nabídka menu, tvořeny seznamem typu „ul“ s položkami typu „li“. Každá položka obsahuje název skupiny grafů, které jsou obsaženy v příslušné kartě. Je zde tedy vytvořen element typu „a“. Tento element obsahuje atribut „aria-controls“, který určuje element typu „div“ se stejným obsahem atributu typu „id“. Dále je zde důležitý atribut typu „onclick“, který opět odkazuje do JavaScriptové knihovny, která vykreslí příslušný graf. Pod tímto seznamem je definován druhý vnořený element typu „div“, který obsahuje další vnořené elementy typu „div“, na které odkazuje atribut typu „aria-controls“, v odkazech seznamu záložek. V každém elementu typu „div“ je vytvořeno plátno typu „span“ s atributem typu „id“, podle kterého je plátno identifikováno v JavaScriptové knihovně.

Po uzavření všech párových elementů, je na konci element typu „script“. V tomto elementu jsou volány funkce, které automaticky po vykreslení stránky provedou doplnění hodnot do panelů a vykreslí graf v aktivní kartě. Automaticky je zvolena první karta.

Stránka s chybami – error.html

Stránka s chybami obsahuje dva hlavní panely. První panel je společný s hlavní stránkou „index.html“. Jedná se o informace o testu (adresa stránky, doba trvání testu, začátek a konec testu). Tento panel je programován stejným způsobem.

Druhý panel je tvořen elementem typu „div“ s druhým vnořeným elementem typu „div“ s atributem typu „class“ s hodnotou „panel-body“. V tomto elementu se nachází plátno pro graf typu „canvas“, kde je definována pevná výška a šířka na 300px. Pod tímto plátnem se nachází element typu „div“. V tomto elementu jsou čtyři elementy typu „span“. V prvním a třetím elementu se nachází slovní definice následující hodnoty. A v druhém a čtvrtém číselná hodnota. Tyto hodnoty vyjadřují procentuální podíl úspěšných a neúspěšných dotazů během testu. Poslední element v hlavním panelu je typu „table“. Do tohoto elementu je opět pomocí JavaScriptové knihovny vykreslena tabulka s nejčastějšími chybami, které během testu nastaly.

Stejně jako u hlavní stránky se na konci těla této stránky nachází volání funkcí, které vyplní panely hodnotami a vykreslí graf s tabulkou.

Stránka se statistikami – statistics.html

I u poslední stránky se nachází panel s informacemi o testu. Pod tímto panelem jsou tři elementy typu „div“. První obsahuje vnořený element typu „table“. Tato tabulka je stejně jako všechny tabulky vykreslována JavaScriptovou knihovnou. Obsahuje kompletní statistiky. Tyto statistiky pocházejí z objektu dataCollector, který je popsán v sekci Struktura vlastního objektu. Následuje další element typu „div“.

V tomto elementu je vnořen další element typu „tableL“. V této tabulce jsou obsaženy percentily. Poslední prvek v panelu je plátno typu „canvas“, na které je vykreslován graf s percentily. Na konci těla stránky jsou opět volány JavaScriptové funkce.

Stránka pro výsledky DDOS útoku a serveru – ddos.html a server.html

Na těchto stránkách se nachází pouze oznámení o pozdější implementaci funkcionality. Tyto stránky byly vytvořeny na základě přání zadavatele práce.

3.2.4 Testování zobrazení stránky

Webová stránka byla testována v následujících prohlížečích: Google Chrome verze 58.0, Opera verze 45.0, Mozilla Firefox verze 53.0, Vivaldi verze 1.9 a Microsoft Edge ve verzi 40.15063.0.0. Na prohlížečích typu Internet Explorer není správná funkčnost zaručena. Testování na této platformě bylo prováděno, avšak neúspěšně z důvodu blokování skriptovacích souborů, kde i přes povolení se skript neprovedl. Z důvodu konce podpory pro tuto platformu (Internet Explorer) ze strany výrobce ke dni 12. ledna 2016 nebyla další optimalizace prováděna. O konci podpory prohlížečů Internet Explorer více zde: [11]. Další testy byly provedeny v rámci rozlišení obrazovky monitoru. Stránky byly úspěšně vykresleny na monitorech s rozlišením 2560x1440, 1920x1080 a 1680x1050.

3.2.5 Výkonná část

Výkonnou částí se rozumí knihovny psané v jazyce JavaSrcipt. Tyto knihovny se starají o práci s daty, která jsou předána modulem z programu JMeter. Předaná data jsou v knihovnách roztříděna a následně vykreslena do tabulek a grafů.

Systém předávání dat z programu JMeter

Generování šablony využívá dvou metod. První metodou je přímá úprava souboru s JavaScriptovými funkcemi (soubor „functions.js“). Celkový systém zápisu výsledků je popsán v sekci Práce s daty. Druhá metoda spočívá v umístění volání funkce pro funkci Dashboard. Ve výpisu 3.4 je umístěna ukázka volání funkce, pro vygenerování výsledků pro graf „Response Time Distribution“.

Výpis 3.4: Volání funkce pro předání dat Response Time Distribution.

```
1 \${responseTimeDistribution!"{}"}
```

Funkce Dashboard toto volání ze souboru smaže a na jeho místě zanechá data ve formátu JSON.

3.2.6 Popis výkonné části

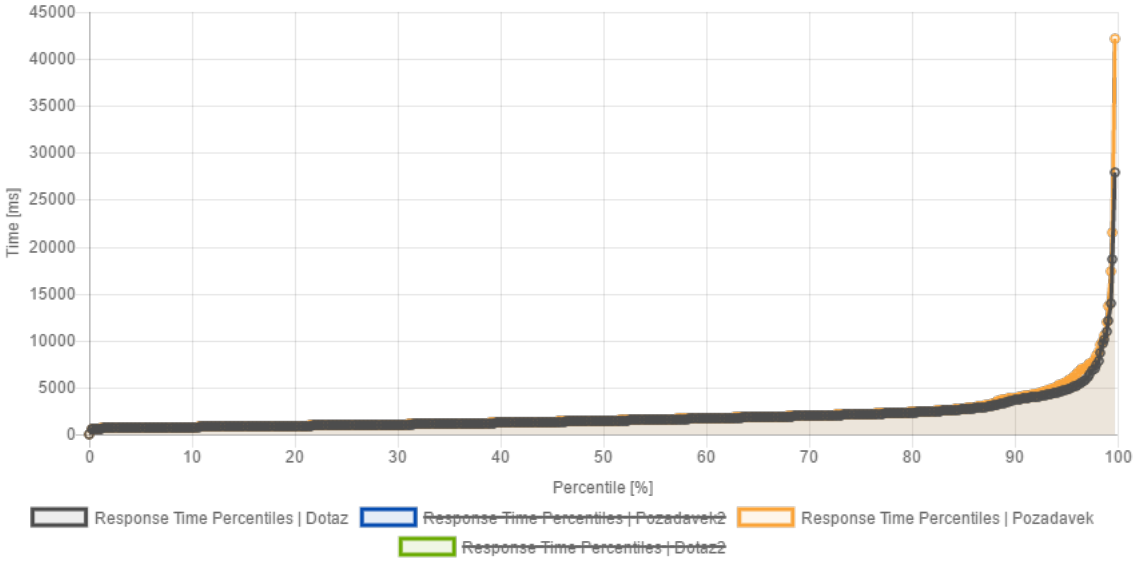
Všechny funkce, které provádí výkonná část jsou uloženy v souboru „functions.js“. Další JavaScriptové soubory jsou funkcionality třetích stran, které jsou v této práci využity.

Jako první jsou v JavaScriptovém souboru definovány proměnné pro data grafů. Pro lepší práci jsou následně proměnné, které mají závislost na čase, uloženy do pole. Toto pole je nazvané „dataBasedOnTime“. Po deklaraci tohoto pole následuje vytvoření prázdného objektu „lineChartData“, který slouží pro předání dat do grafu. V tomto objektu lze definovat data na ose x a na ose y. Následuje vytvoření objektu s názvem „canvasData“. Tento objekt obsahuje předchozí objekt „linearChartData“. Navíc jsou v tomto objektu definovány volby nastavení. Jsou to: typ vykreslení grafu (linka), povolení responzivnosti a nastavení chování při najetí kurzorem na bod v grafu. Po této sadě nastavení následuje nastavení os. Každou osu lze nastavit zvlášť. V nastavení osy jsou parametry pro definování: typu osy (lineární), zda ji zobrazit (ano) a její pozici (vlevo pro osu y a dole pro osu x). Toto jsou základní objekty potřebné pro správnou funkčnost knihovny ChartJS.

Následuje první funkce `drawGraph()`, která je volaná při vyvolání karty s grafem na hlavní stránce („index.html“). Vstupním parametrem je proměnná „typeOfGraph“. Do této proměnné je vložen název grafu, který bude vykreslen do příslušné karty. Celou funkci tvoří přepínač „switch“. Jako výraz pro tento přepínač je použit parametr funkce. Na základě jeho obsahu je vybráno návěstí s odpovídajícím názvem. Funkce v této návěstí zavolá příslušnou funkci pro volaný typ grafu. Po vykonání funkce a vytvoření příslušných dat pro graf je přidán příznak „_canvas“ do parametru. Tento parametr je následně použit v proměnné „ctx“, která na stránce najde příslušné plátno (element „canvas“) a je předána knihovně ChartJS. Tato knihovna vykreslí graf do příslušného plátna. Pokud v přepínači není nalezeno příslušné návěstí, je automaticky zvoleno návěstí „default“ a je vypsána chybová hláška o nenalezení typu grafu do konzole prohlížeče.

Druhá funkce má název `createVariables()`. Tato funkce je volaná funkcí, která plní data do objektů určených pro generování grafů. Tato funkce má za úkol připravit pole se jménem „dataSets“. Toto pole obsahuje objekty s daty pro jednotlivé grafy, které budou vykresleny na jedno plátno. V objektu se nachází barva linky, barva plochy pod linkou, název grafu, maximální hodnota osy, minimální hodnota a stavovou proměnnou, zdali má být oblast pod linkou vybarvena. Dále se v objektu nachází pole pro data grafu. Po nastavení všech hodnot je objekt zapsán do pole. Počet objektů závisí na počtu grafů v parametru funkce. Tento parametr je pole objektů, kde každý objekt reprezentuje jeden graf. Jeden graf může obsahovat další grafy. Tím se rozumí, že například u grafu se jménem „Percentiles over Time“ jsou

další grafy. Každý graf odpovídá jednomu požadavku HTTP. Tyto grafy v jednom grafu lze vidět na obr. 3.5, kde jsou dva ze čtyř grafů zobrazeny.



Obr. 3.5: Graf s percentilem času odpovědi.

Následující funkce s názvem `genPercentileChart()` generuje graf s percentily. Tento graf je možné vidět na obr. 3.5. Na začátku funkce jsou připraveny pomocné proměnné. Dále jsou definovány osy. Jsou definovány do předpřipravených objektů ze začátku. V objektu osy se přidá její název, typ, pozice a stav zobrazení (zobrazeno). Po tomto nastavení přijde na řadu plnění pole daty. Program JMeter vygeneroval strukturu vnořených polí, která obsahují data pro jednotlivé grafy. Tato pole jsou pomocí cyklů for procházena a data jsou následně plněna do pole pro graf. Využívá se pomocných proměnných, které byly definovány nad cykly for. Protože jazyk JavaScript neobsahuje cyklus foreach, byl tento cyklus vytvořen jako funkce, která je v plnění dat využívána. Další funkce s názvy `genLatencyVsRequestChart()`, `genTimeVSThreadsChart()`, `genresponseTimeDistributionChart()` a `fillLineChartData()` fungují stejně a pouze jsou poupraveny pro potřeby jednotlivých grafů. Jsou to například názvy os, formátování časových razítek pomocí JavaScriptové knihovny MomentJS na požadovaný formát času, atd...

Další funkcí je funkce `drawPieGraph`. Tato funkce generuje koláčový graf, který se nachází na stránce s chybami („errors.html“). V této funkci je vytváření grafu obdobné jako u předchozích funkcí. Pouze je typ grafu „pie“. A liší se plnění daty, kde se nevyužívá cyklu for, ale pole je plněno přímo. Hodnoty pro graf pocházejí z objektu „summaryData“. Hodnoty se převedou na typ String a následně se použije prvních pět míst proměnné String. Tato data se následně použijí pro graf.

Po funkcích generujících graf, následují funkce pro generování barev v grafech. O generování se stará funkce `getRandomColor()`. V této funkci je pole, které obsahuje základní sadu barev. Toto pole obsahuje jedenáct barev, které jsou ve formátu Červená, Zelená a Modrá, RGB (red, green, blue) a následně jsou kvůli průhlednosti převáděny do formátu Červená, Zelená, Modrá a Alfa kanál, RGBA (red, green, blue, alpha). Více o barevných formátech používaných na webu zde [3]. Po funkci generování barev s alfa kanálem následuje funkce, která alfa kanál nastaví na hodnotu 1, tj. 100% z hodnoty 0,1, kterou vrací předešlá funkce. Funkce pro generování barev se využívají ve funkcích pro generování grafů.

Poslední graf, který je na webové stránce, je graf s percentily. O vykreslení tohoto grafu se stará funkce `drawBarGraph()`.

Generování tabulek a popisů

První funkce, která generuje tabulku, má název `drawErrTable()`. Funkce generuje tabulku nejčastějších chyb na stránce s chybami („errors.html“). Tato funkce obsahuje parametr, který funkci předává data tabulky. Jako první jsou data rozebrána (rozparsována) a uložena do pomocné proměnné. Jako u předešlých funkcí rozebráním dat vznikne objekt obsahující pole. Opět pomocí cyklů for jsou generovány řádky a sloupce tabulky. Po vygenerování je tabulka vykreslena do elementu s argumentem „id“ s hodnotou „mainTable“.

Po funkci pro generování tabulky s chybami následuje funkce s názvem `generateTime()` pro vyplnění hodnot času v horním panelu. Funkce obsahuje dva parametry. První je pro počáteční čas a druhý pro konečný čas. Oba formáty jsou textového charakteru. Protože jde o hodnoty vygenerované programem JMeter, je zapotřebí odstranit nežádoucí znaky. V tomto případě se jedná o uvozovky. Po odstranění dojde k vypsání do panelu. Následuje převod na formát data. Oba časy se od sebe odečtou a vznikne celkový čas testu. Ten se opět převede to textové formy a vypíše se do panelu.

Další funkce má název `fillData()`. Tato funkce vypisuje hodnoty do levého panelu na hlavní stránce. Následuje funkce `fillURL()`. Tato funkce vypisuje testovanou webovou stránku do horního panelu. Jako všechny tyto funkce vypisuje text pomocí identifikátoru id v elementu. Tato funkce navíc přidá textu charakter odkazu. Funkce `fillErrStats()` vyplňuje hodnoty vedle koláčového grafu. Jako pomocná funkce slouží funkce `Round()`. Tato funkce slouží k zaokrouhlování. Využívá se ve funkcích při plnění pole dat. Má dva vstupní parametry. První parametr „num“ je pro číslo, které bude zaokrouhleno a druhý parametr „precision“ je pro počet desetinných míst, na která se má číslo zaokrouhlit.

Následuje funkce s názvem `createStatisticsTable()`, jež vykreslí tabulku se

souhrnnou statistikou na stránce se statistikami („statistics.html“). Tato funkce má v poli „titles“ definované záhlaví tabulky. Následuje generování tabulky podobně jako u funkce `drawErrTable()`. Opět se využívá cyklů for, které postupně procházejí pole hodnot a vypisují jej do tabulky. Pouze u posledního sloupce je vypsáno pole adres, se kterými bylo během testu komunikováno. To bylo veřešeno pomocí tlačítka, které adresy skryje. Po kliknutí na tlačítko se adresy zobrazí.

Pod tabulkou se souhrnnou statistikou se nachází tabulka percentilů. Tuto tabulku vykresluje další funkce s názvem `createPercentilTable()`. Tato funkce funguje stejně jako předchozí. Pouze je opět poupravena pro potřeby tabulky s percentily.

Pro podrobnou ukázku funkcionality je přiložena šablona stránky a vygenerovaná ukázková stránka v příloze na CD.

4 ZÁVĚR

Cílem práce bylo provedení analýzy dostupných řešení pro zátěžové testování a následný návrh a vývoj systému, který by automaticky generoval report zátěžového testu ve formě webové stránky. Celý systém je implementován do nástroje JMeter jako přídavný modul.

První kapitola se věnuje teorii zátěžového testování. Úvodní odstavec této kapitoly obsahuje krátké seznámení se zátěžovým testováním. Po úvodu pokračuje kapitola seznamem typů zátěžových testů. Každý test je popsán a je uveden jeho účel. Poslední částí kapitoly o zátěžovém testování jsou základní vstupní i výstupní parametry zátěžových testů.

Druhá kapitola je zaměřená na analýzu dostupných řešení na Internetu. Tato řešení jsou popsána od založení projektu, po prezentaci výsledků.V této kapitole byly analyzovány tři webová komerční řešení (Blazemeter, LoadImpact a Apica) a jedno řešení založené na otevřeném zdrojovém kódu, jenž je nutné provozovat na vlastním zařízení. Jedná se o nástroj JMeter. Na základě analýzy struktury výsledků, byla jednotlivá řešení porovnána. Kritérií porovnání byly forma podání výsledků, možnosti generování do souboru a funkce pro porovnání s předešlými testy. Poslední důležitý aspekt byl interaktivnost grafů.

Poslední kapitola popisuje návrh modulu pro program JMeter. Je rozdělena na dvě podkapitoly. První podkapitola se věnuje popisu programového kódu v jazyce Java a funkcí, které modul využívá z projektu JMeter. Druhá podkapitola popisuje programový kód šablony a postup návrhu webové stránky. Tento návrh vycházel z poznatků získaných v první části, věnující se analýze dostupných nástrojů. Následuje popis kódu webové stránky. Poslední popisovaný kód zpracovává data předaná stránce modulem. Modul zpracovává výsledky z testu, který je programem JMeter prováděn. Výsledky jsou následně pomocí funkcionality `Dashboard` zpracovány a převedeny do podoby webové stránky. Protože funkcionalita `Dashboard` neobsahuje všechna potřebná data, která jsou ve výsledcích očekávána, je modul rozšířením pro tuto funkcionalitu. Vygenerovaná webová stránka je naprogramována v jazyce HTML, CSS a JavaScript. Pro vytvoření webové stránky je zvolen aplikační rámec Bootstrap. Webová stránka obsahuje grafy, které jsou generovány aplikačním rámcem CharJS.

Výsledkem této práce je modul, který rozšiřuje funkcionalitu programu JMeter. Oproti dostupné funkci `Dashboard` obsahuje výsledky jako jsou například: seznam chyb, latence, výpočet percentilu, průměrnou dobu připojení k testovanému serveru, seznam testovaných webových adres a další. Do vytvořeného modulu budou v budoucnu začleněny další funkcionality. Například se bude jednat o statistiky serverové části a statistiky DDoS, pro které jsou v šabloně již nachystané stránky.

LITERATURA

[1] BLAZEMETER.COM *Znalostní báze Blazemeter.* [online]. Mountain View, California: Blazemeter, 2016 [cit. 19. 10. 2016]. Dostupné z URL: <https://guide.blazemeter.com/hc/en-us/articles/207421515-Request-Statistics-Report/>.

[2] *Bootstrap.* About Bootstrap [online]. San Francisco, CA: Bootstrap, 2011 [cit. 05. 05. 2017]. Dostupné z URL: <http://getbootstrap.com/about/>.

[3] *CSS Legal Color Values.* W3Schools [online]. Internet: W3Schools, 2017 [cit. 25. 05. 2017]. Dostupné z URL: <https://www.w3schools.com/cssref/css_colors_legal.asp>.

[4] ECKEL, Bruce. *Thinking in Java. 4th ed.* Upper Saddle River, NJ: Prentice Hall, c2006. ISBN 01-318-7248-6.

[5] F. N. Bernardo Gois, P. P. M. de Farias, A. L. V. Coelho and T. M. Barbosa, *Improving stress search based testing using a hybrid metaheuristic approach,* 2016 XLII Latin American Computing Conference (CLEI), Valparaíso, 2016, pp. 1-11.

[6] *ChartJS.* API Documentation [online]. Internet: ChartJS, 2016 [cit. 05. 05. 2017]. Dostupné z URL: <http://www.chartjs.org/docs/>.

[7] JMETER.APACHE.COM *JMeter. Apache JMeter™* [online]. Los Angeles: Apache, 2016 [cit. 21. 11. 2016]. Dostupné z URL: <http://jmeter.apache.org/>.

[8] *List of web testing tools.* In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2017 [cit. 20. 05. 2017]. Dostupné z URL: <https://en.wikipedia.org/wiki/List_of_web_testing_tools>.

[9] *Load Testing Tools. Software Testing Tools* [online]. Berlin: GUROCK SOFTWARE, 2017 [cit. 01. 06. 2017]. Dostupné z URL: <http://www.testingtools.com/load-testing/>.

[10] *MomentJS* [online]. Palo Alto: MomentJS, 2011 [cit. 24. 05. 2017]. Dostupné z URL: <http://momentjs.com/>.

[11] *Support for older versions of Internet Explorer ended.* Microsoft [online]. Redmond: Microsoft Corporation, 2016 [cit. 24. 05. 2017]. Dostupné z URL: <https://www.microsoft.com/en-us/windowsforbusiness/end-of-ie-support>.

[12] *The Java EE 6 Tutorial.* Oracle Docs [online]. Redwood City: Oracle and/or its affiliates, 2013 [cit. 24. 05. 2017]. Dostupné z URL: <http://docs.oracle.com/javaee/6/tutorial/doc/gjbbp.html>.

[13] *Zátěžové testování webových aplikací: Úvod.* In: Etnetera.cz [online]. Praha: Etnetera, 2014 [cit. 05. 12. 2016]. Dostupné z URL: <http://www.etnetera.cz/archiv/773-tech_life/tech_life_140117_zatezove_testovani_webovych_aplikaci.html>.

[14] *Znalostní báze Apica.* [online]. Stockholm: Apica, 2016 [cit. 07. 11. 2016]. Dostupné z URL: <http://kb.apicasystem.com/display/LTP/LoadTest+Portal+Home>.

[15] *Znalostní báze Load Impact.* [online]. Stockholm: Load Impact, 2016 [cit. 24. 10. 2016]. Dostupné z URL: <http://support.loadimpact.com/>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

CPU	Centrální procesorová jednotka – Central processing unit
CSS	Kaskádové styly – Cascading Style Sheets
CSV	Hodnoty oddělené čárkami – Comma-separated values
DDoS	Distribuované odepření služby – Distributed denial of service
DoS	Odepření služby – Denial of service
DNS	Systém doménových jmen – Domain Name System
HTML	Hypertextový značkovací jazyk – HyperText Markup Language
HTTP	Hypertextový transportní protokol – Hypertext Transfer Protocol
IP	Internet Protocol
JSON	JavaScript Object Notation
KPI	Ukazatele výkonnosti – Key Performance Indicator
LTE	3GPP Long Term Evolution
RAM	Random-access Memory
RGB	Červená, Zelená a Modrá – Red, Green, Blue
RGBA	Červená, Zelená, Modrá a Alfa kanál – Red, Green, Blue, Alpha
SSL	Vrstva bezpečných socketů – Secure Sockets Layer
TCP	Transmission Control Protocol
URL	Jednotná adresa zdroje – Uniform Resource Locator
XML	eXtensible Markup Language

5 OBSAH PŘILOŽENÉHO CD

Přiložené CD obsahuje elektronickou verzi práce ve formátu PDF. Dále se na CD nachází zdrojový kód modulu pro program JMeter a šablona webové stránky. Adresářová struktura přiloženého CD:

- `/` ..... kořenový adresář přiloženého CD
  - `dokumentace.pdf` ..... Elektronická verze dokumentace
  - `report-template` ..... Šablona webové stránky
    - `content`
      - `css`
        - `bootstrap.min.css`
        - `style.css`
        - `style2.css`
      - `fonts`
        - `OpenSans`
          - `OpenSans-Regular.woff`
        - `VUT`
          - `Vafle-Regular-web.woff`
      - `chart`
        - `Chart.bundle.js`
        - `Chart.js`
        - `utils.js`
      - `img`
        - `gity`
          - `logo-gity.svg`
        - `vut`
          - `logo-vut.svg`
      - `js`
        - `bootstrap.min.js`
        - `functions.js.fmkr`
        - `jquery-3.1.1.min.js`
        - `moment.js`
      - `pages`
        - `ddos.html.fmkr`
        - `error.html.fmkr`
        - `server.html.fmkr`
        - `statistics.html.fmkr`
    - `index.html.fmkr`
  - `modul` ..... Zdrojové soubory modulu WebGenerator
    - `WebGenerator.java`
    - `WebGenerator.jar`
  - `readme.txt` ..... Postup pro zprovoznění modulu