Zápočtová úloha z předmětu KIV/ZSWI

DOKUMENT SPECIFIKACE POŽADAVKŮ

6. května 2015

Název týmu: **CrossCafe team**

Členové týmu:

Luboš Hubáček (vedoucí týmu)	**lubos.hubacek@diginex.cz**
Ondřej Pittl	**ondrej.pittl@gmail.com**
Jiří Homolka	**jiri.homolka22@gmail.com**
Jan Kohlíček	**kohlicejan@gmail.com**
Petr Kozler	**pkozler@students.zcu.cz**

JAVA aplikace pro monitoring běhu komponent

DOKUMENT SPECIFIKACE POŽADAVKŮ

Verze <1.0>

HISTORIE DOKUMENTU

Datum	Verze	Popis	Autor
9.3.2015	1.0	Vytvoření dokumentu	Luboš Hubáček, Petr Kozler

Obsah dokumentu

1 Úvod

1.1 Předmět specifikace

Naprogramujte aplikaci v jazyce Java, která bude monitorovat stav daných komponent. Ke zjišťování stavu komponent použijte výměnu dat, předávaných ve formátu JSON, pomocí jednoduchých HTTP příkazů. Získané informace poté ukládejte do předpřipravených souborů (pro každou sledovanou instanci jeden soubor) s konstantní velikostí (ukládání bude probíhat způsobem, který se označuje jako tzv. rolling file). Pro vytváření souborů použijte knihovnu Log4J ve verzi 2. Součástí programu bude i grafické uživatelské rozhraní implementované pomocí knihovny JavaFX. GUI bude obsahovat jednak strom obsahující instance komponent a také textové pole, které poslouží pro zobrazení výstupů z jednotlivých instancí filtrovaných označením příslušných položek ve stromu.

Příklad URL pro získávání dat (komponent): http://peerfile.eu:3000/api/mon/memory_info

1.2 Typografické konvence

Při vytváření použijte běžné typografické konvence. Jedná se zejména například o psaní zdrojového kódu, programátorské dokumentace a rozhraní programu v anglickém jazyce. Tímto bude částečně zašťítěna kompatibilta s konvencemi využívanými u zadavatele.

1.3 Cílové publikum

Specifikace je určená zadavateli projektu a týmu, který jí zpracovává. Kromě zadavatele bude specifikaci číst ještě několik jeho kolegů, kteří později tuto aplikaci začlení do jiného většího systému.

1.4 Rozsah projektu

Cílem práce je vytvořit monitorovací aplikaci, která bude přehledně logovat data o stavu systémů do souborů, pro archivaci těchto dat.

1.5 Odkazy

Unicorn. Dokumentace Monitoring REST API. *Google dokumenty.* [online]. 2014 [cit. 2015-03-09]. Dostupné z: *http://bit.ly/1DeFsnZ*

CrossCafe team. ZSWI monitoring. *GitHub.* [online]. 2015 [cit. 2015-03-09]. Dostupné z: *http://github.com/Sekiphp/ZSWI_monitoring*

2 Obecný popis

2.1 Kontext systému

Jedná se o novou část již fungujícího stávajícího produktu, jejíž úlohou bude čtení dat vytvářených jinými částmi systému a unmarshalling z formátu JSON do snadno čitelné podoby.
Nakreslete jednoduchý diagram znárodňující hlavní komponenty systému, vztahy mezi podsystémy a externí rozhraní.

2.2 Funkce produktu

Tato aplikace bude používána ke zjišťování aktuálního stavu daných komponent a ukládání výstupu do rotujících souborů. Bude poskytovat jednoduché grafické uživatelské rozhraní pro pohodlné ovládání programu.
Připojte obrázek hlavních skupin požadavků a vztahů mezi nimi (~~diagram datových toků, diagram případů užití~~, diagram tříd).

2.3 Třídy uživatelů

Aplikace je určená pro technicky zdatné uživatele, jelikož se jedná o podpůrnou monitorovací službu. Všichni uživatelé aplikace budou na využívat stejného rozhraní – žádné administrátorské účty s vyššími oprávněními nejsou vyžadovány.

2.4 Provozní prostředí

Aplikace poběží na Linuxovém serveru (předpoklad).
Popište prostředí, ve kterém systém poběží – včetně hardwaru, typu a verze operačního systému, zeměpisného umístění uživatelů, serverů a databází. Uveďte seznam všech softwarových komponent a systémů, se kterými musí systém spolupracovat.

2.5 Omezení návrhu a implementace

Aplikace musí být naprogramována v jazyce Java. K vytvoření GUI musí být použita knihovna JavaFX. Pro logování knihovna Log4J 2 a pro výměnu dat framework Spring WS. Při výměně dat bude nutné pracovat s formátem JSON. Pro verzování a synchronizaci vývoje mezi členy týmu musí být použita služba GitHub a pro správu závislostí programu nástroj Maven. Na použitá IDE nejsou kladeny žádné zvláštní požadavky.

2.6 Uživatelská dokumentace

Uživatelská dokumentace bude realizována pomocí wiki na stránkách projektu na GitHubu (odkaz v kapitole 1.5) a pomocí textové dokumentace ve formátu PDF.

2.7 Předpoklady a závislosti

Tato aplikace bude napojena na stávající komponenty většího systému. Funkce programu je tedy závislá na funkci těchto komponent (zejména Peerfile backend (PF)). Pokud dojde k výpadku služby PF tento program nebude moci fungovat – zaznamená se výpadek systému.

3 Funkce systému

3.1 Webservice klient

3.1.1 Popis a priorita

Priorita: vysoká

3.1.2 Události a odpovědi

Událost: uživatel označí instanci komponenty ve stromě
Odpověď: program odešle příslušný požadavek pro získání dat

3.1.3 Funkční požadavky

Pokud se nepodaří odeslat požadavek, program zobrazí chybovou hlášku a zaloguje událost do souboru a vypíše na terminál.

3.2 Parsování JSON response

3.2.1 Popis a priorita

Priorita: vysoká

3.2.2 Události a odpovědi

Událost: je přijata odpověď ve formátu JSON
Odpověď: program provede parsování přijatých dat

3.2.3 Funkční požadavky

V případě neúspěšné operace program zareaguje obdobně jako v předchozím případě (bod 3.1.3).

3.3 Ukládání do rotujících souborů

3.1.1 Popis a priorita

Priorita: střední

3.1.2 Události a odpovědi

Událost: přijatá data ve formátu JSON byla naparsována a uložena do paměti
Odpověď: program zaloguje data v textové podobě do rotujícího souboru

3.1.3 Funkční požadavky

V případě neúspěchu při zápisu do souboru program zobrazí příslušnou chybovou hlášku.

3.4 Prezentace dat v GUI

3.1.1 Popis a priorita

Priorita: nízká

3.1.2 Události a odpovědi

Událost: přijatá data ve formátu JSON byla naparsována a uložena do paměti
Odpověď: program zobrazí přijatá data v čitelné podobě do textového pole

3.1.3 Funkční požadavky

U této části aplikace se neočekávají žádné chybové stavy.

4 Požadavky na vnější rozhraní

4.1 Uživatelská rozhraní

GUI aplikace bude sestávat z jediného hlavního okna a případně jednoduchých dialogových oken. Na vizuální podobu písma či ovládacích prvků nejsou kladeny žádné zvláštní požadavky, pravděpodobně budou použita výchozí nastavení komponent knihovny Java FX. Není vyžadována lokalizace softwaru ani možnosti usnadnění ovládání pro handicapované.

4.2 Hardwarová rozhraní

V této aplikaci není třeba HW rozhraní řešit.

4.3 Softwarová rozhraní

Pro přijimání dat bude využito rozhraní instancí PF (více o PF v sekci odkazy – dokumentace v daném dokumentu). O výměnu dat se v tomto případě postará protokol JSON za pomoci HTTP hlaviček. Toto je jediný způsob, jak bude program přijímat data.

API pro tento program, které by usnadnilo komunikaci s případnými dalšími programy není vyžadováno.

4.4 Komunikační rohraní

ONDRO, KOUKNEŠ NA TO? ASI POPSAT JSON A PF

Popište požadavky na všechny komunikační funkce, které bude systém používat (e-mail, webový prohlížeč, protokoly, elektronické formuláře, ad.). Popište formát pro výměnu zpráv, zabezpečení komunikačních kanálů, šifrování, přenosové rychlosti, synchronizační mechanismy.

5 Další mimofunkční požadavky

5.1 Výkonostní požadavky

Uveďte konkrétní požadavky na výkon jednotlivých funkcí systému a vysvětlete důvody, které k nim vedly. Buďte co nejpřesnější (např. „na jednouživatelském PC s procesorem Intel Pentium Core 2 Duo o taktovací frekvenci 3,16 GHz a OS Windows XP s alespoň 60% volných systémových prostředků bude 95% databázových dotazů do registru vyřízeno do dvou sekund)

5.2 Bezpečtnostní požadavky

Nejedná se o business critical aplikaci, nicméně tato aplikace monitoruje business critical procesy, proto je vyžadována její 100% spolehlivost a v přípravě havárie by měla být provedena brzská oprava, ideálně proveditelná do jedné hodiny od zaznamenaného výpadku.

5.3 Kvalitativní požadavky

- požadavky na snadnost používání – nejsou
- požadavky na spolehlivost – je vyžadována na 100%
- požadavky na robustnost – oprava po havárii musí proběhnout do 1 hodiny
- požadavky na udržitelnost – zdrojový kód a programátorská dokumentace (Javadoc) budou psány v anglickém jazyce; budou dodržovány konvence běžné v programovacím jazyce Java (například camel case)

5.4 Ostatní požadavky

Na aplikaci nejsou kladeny žádné další zvláštní požadavky.

6 Dodatek A: slovníček pojmů

PeerFile (=PF) -

REST API –

JSON – způsob přenosu dat nezávislý na počítačové platformě. Vstupem je libovolná datová struktura, výstupem je vždy řetězec. Složitost hierarchie vstupní proměnné není teoreticky nijak omezena – JSON může mít různě složitou strukturu.

Rolling File – Soubor s konstatní velikostí, kde dochází k odmazání starších dat, když by mělo dojít k překročení kapacity

7 Dodatek B: seznam úkolů

Zde budeme průběžně doplňovat seznam všech úkolů. Asktuální úkoly jsou:

- načíst data z PF instancí
- zprovoznit Maven
- logování pomocí Log4J 2