Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Systém workflow v EEG/ERP Portálu

Plzeň 2013 Radek Mrvec

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 9. května 2013

Radek Mrvec

Poděkování

Rád bych poděkoval panu Ing. Petru Ježkovi PhD. za vedení mé práce, komunikaci, okamžité odpovědi a trpělivost s mými neustálými dotazy.

Abstract

This work is focused on development of a work flow system for running methods for signal processing. First it tries to find suitable way to implement work flow system in e-neuroscience. The work analyses other applications and chooses the most suitable one for the integration of the work flow system. When research is done, it implements all necessary changes to be able to run work flow data processing from our application. Because of complexity of workflows a prototype implementation is realized. This work serves as an initial proposal of workflow system in the EEG/ERP domain.

Obsah

1 Úvod

Cílem této práce je návrh a implementace workflow systému v EEG/ERP Portálu. S workflow systémy se setkáváme v běžném životě velmi často a patří mezi nedílnou součást informačních systémů většiny podniků. Pomocí workflow systému můžeme například jasně definovat oběh a zpracování dokumentů v rámci různě velkých skupin.

Tato práce si klade za cíl nalezení optimálního využití workflow při zpracování dat získaných měřením EEG/ERP. Tyto data je možné v rámci různých experimentů dále zpracovávat matematickými metodami. Při použití workflow systému bude umožněno uživatelům definovat více metod a dat v jednom zpracování. Dále uživatel bude mít možnost vytvářet komplexní sled kroků, které ho jednoduchým způsobem dovedou k požadovanému cíli. Bez použití workflow by uživatel musel jednotlivá data zpracovávat postupně, což by mu zabralo mnohem více času a stálo více usilí.

Po nalezení a návrhu optimálního workflow systému bude tento systém implementován a následně integrován do současného EEG/ERP Portálu, který je provozován Západočeskou Univerzitou v Plzni, Katedrou informatiky a výpočetní techniky.

Vzhledem k tomu, že problematika workflow je velmi komplexní, bude implementace provedena ve formě funkčního prototypu, který bude dále rozvíjen výzkumnou skupinou, zabývající se vývojem a implementací EEG/ERP Portálu.

V první části práce se nejdříve seznámíme s problematikou EEG/ERP, webovou aplikací EEGbase a technologiiemi používanými při vývoji webových aplikací. V druhé části analyzujeme konkurenční aplikace, které by bylo možné využít pro správu workflow systému. V další části ná základě předchozí analýzy navrhneme systém workflow a jeho následnou integraci do aplikace EEGbase. V poslední části naší práce se budeme věnovat samotné implementaci navrhovaného řešení.

2 Seznámení s EEG/ERP Portálem

V této části práce se seznámíme s projektem EEG/ERP Portálu a technologiemi, které jsou používané při vývoji této webové aplikace.

2.1 Úvod do problematiky EEG/ERP

Nejdříve se pokusíme stručně představit problematiku získání EEG/ERP dat a definovat tyto pojmy.

2.1.1 Co je to EEG

Tento termín definuje záznam časové změny elektrického potenciálu způsobeného mozkovou aktivitou. Tento záznam se získává pomocí přístroje zvaného elektroencefalograf. Přístroj snímá elektrické potenciály pomocí elektrod připevněných na povrchu hlavy. Tyto informace po zesílení zpracovává a zapisuje křivku na papír nebo na obrazovku. Tvar křivky je závislý na aktuální aktivitě mozku, tudíž jinou křivku získáme ve spánku, či při běžné denní aktivitě.

2.1.2 Co je to ERP

Tento termín představuje elektrickou aktivitu mozku jako reakci na podněty zvenčí. Podněty je možné rozdělit do třech základních skupin:

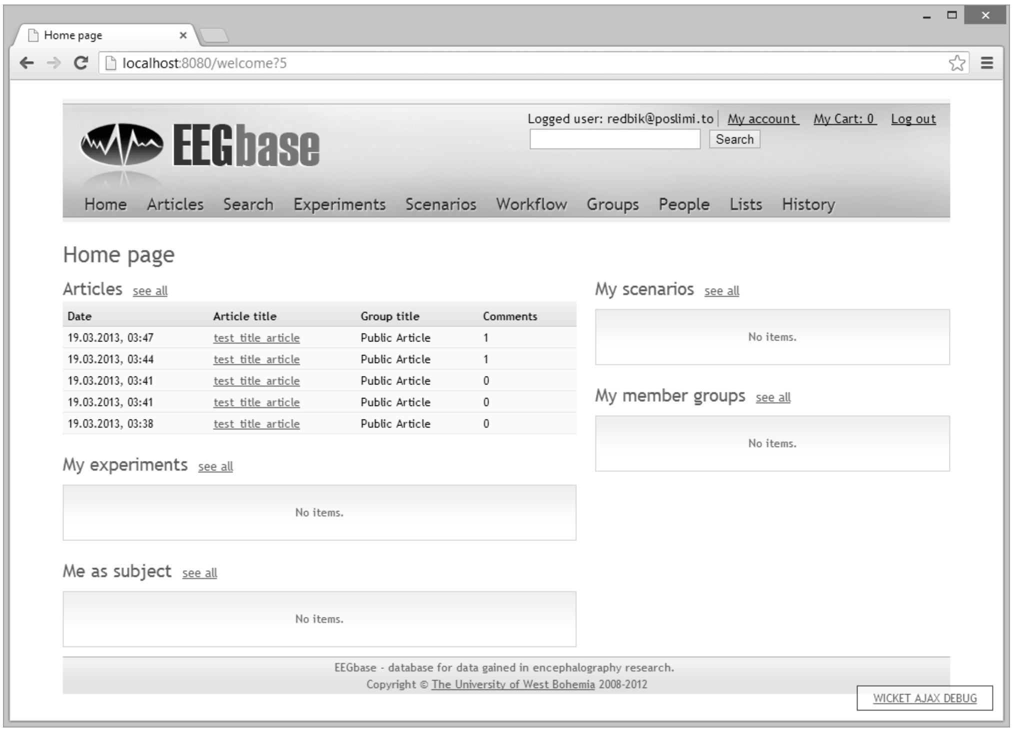
- Zrakové (VEP)
- Sluchové (BAEP)
- Somatosensorické (SEP)

2.2 Popis a funkce aplikace EEGbase

Hlavním cílem aplikace je spravovat data získaná měřením EEG/ERP. Jedná se o otevřenou aplikaci, do které má přístup jakýkoliv registrovaný uživatel. Uživatel zde může zadávat své experimenty, získávat data z již uskutečněných a zveřejněných experimentů. Vzhledem k tomu, že se jedná o webovou aplikaci, tak je aplikace dostupná na jakémkoliv počítači s nainstalovaným internetovým prohlížečem a připojením k síti internet.

Aplikace je zejména určena pro veřejnou vědeckou společnost, zabývající se otázkami EEG/ERP při svých vědeckých výzkumech.

Vzhledem k tomu, že v současné době není znám žádný standardizovaný formát dat získaných z měření EEG/ERP, aplikace umožňuje konverzi různých datových formátů, aby bylo možné data sdílet. Z tohoto důvodu aplikace umožňuje rozšíření o další datové formáty, pomocí vytvoření a přidání nového datového konvertoru. Možnost přidání je dosaženo díky modularitě celého systému.



Obrázek 2.1: *Ukázka prostředí aplikace*

Po přihlášení uživatele je hlavní menu realizováno záložkami v horní části okna. Pod každou záložkou jsou části aplikace týkající se dané záložky [Ježek(2009)].

Aplikace dále umožňuje propojení se sociálními sítěmi, jakými jsou Facebook či LinkedIn. Při propojení je možné se přihlásit k aplikaci pomocí těchto účtů. Mezi další funkce patří např. rozdělení uživatelů do skupin, vyhledávání dat podle nastaveného filtru, fulltextové vyhledávání a další prostředky, na které jsme zvyklí z většiny webových aplikací.

2.3 Technologie ve webových aplikacích

V této části si detailněji přiblížíme nejpoužívanější technolgie, které se používají ve webových aplikacích. Toto přiblížení nám pomůže následně definovat technologie použité v naší aplikaci.

2.3.1 Framework Hibernate

Tento framework poskytuje techniku pro převod dat z objektového modelu do relačního a zpět. Tímto dokáže velmi usnadnit práci s daty nad databází a snížit čas potřebný pro naprogramování podobných metod. Dále framework poskytuje nástroje pro manipulaci s těmito daty, včetně objektově orientovaného jazyka HQL, který na základě objektové notace dokáže uložená data z databáze vybírat v podobě objektů.

Hibernate dále poskytuje mechanismy pro transakční zpracování dat, polymorfní perzistenci, kešování, objektově orientované sestavování dotazů, vlastní datové typy a mnoho dalších funkcí. Samozřejmostí je jednotné API pro práci s řadou databází a databázových serverů[Hib()].

Mapování objektů je možné dvěma způsoby.

- Použití externích XML souborů
- Od verze Java 1.5 použití anotací

Datová vrstva webové aplikace je tvořena sadou tříd, které se starají o manipulaci perzistetních dat nad databází. V naší aplikaci je tato vrstva reprezentována vrstvou přistupující k datům, tzv. Data Access Objects (DAO) vrstvou. Tato vrstva jako jediná komunikuje přímo s Hibernate[Linwood(2010)].

V naší aplikaci je Hibernate konfigurován pomocí XML souborů reprezentujících mapování tříd na tabulky relační databáze. Komunikace mezi Hibernate a databází probíhá pomocí Java Database Conectivity (JDBC).

Listing 2.1: Ukázka namapování tabulky

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
        "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping
           -3.0.dtd">
<hibernate-mapping>
    <class name="cz.zcu.kiv.eegdatabase.data.pojo.SyncChanges"
        schema="EEGTEST" table="SYNC_CHANGES">
        <id name="tableName" type="string">
            <column name="TABLE_NAME" precision="22" scale="0"
                />
            <generator class="increment"/>
        </id>
        <property name="lastChange">
            <column name="LAST_CHANGE" sql-type="timestamp(6)"
                length="11" precision="6" not-null="true"/>
        </property>
    </class>
</hibernate-mapping>
```

2.3.2 Java Server Pages (JSP)

Jedná se o nástroj pro psaní dynamických HTML stránek založený na jazyce Java. Princip je vkládání čistého kódu v Javě přímo do HTML stránky pomocí speciálních značek. Tento kód je poté spuštěn na straně serveru, z čehož zároveň vyplývá, že pokud používáme JSP, musíme zvolit server s podporou tohoto nástroje. Mezi takovéto servery patří např. Apache Tomcat. JSP představuje alternativu k velmi rozšířené technologii PHP a o něco méně používané ASP[Hall(2001)].

Hlavní odlišnosti od PHP je typová kontrola známá z Javy. Další odlišností je kompilace stránek do tzv. servletů, což jsou speciální třídy v jazyce Java, které pak komunikují s webovým serverem. O JSP je tedy možné říci, že jde o nástroj na psaní servletů.

Vzhledem k tomu, že servlety jsou třídy v jazyce Java, mohou využívat další třídy, které nebyly psány pomocí JSP. Tato možnost umožňuje značné urychlení vývoje aplikací a částečné oddělení designu od výkonného kódu. JSP dále umožňují používání tzv. sezení, tedy možnost uchování dat konkrétního uživatele mezi více HTTP požadavky na webovém serveru[Kothagal(a)].

2.3.3 Framework Wicket

Tento framework je alternativou k již dříve zmíněné technologii JSP. Hlavní ideou tohoto frameworku je oddělení HTML kódu od Java kódu. Při psaní aplikace se používá validní HTML kód, kde pomocí parametrů tagů můžeme propojit daný element s Java kódem. Tento framework se svým použitím velmi podobá psaní aplikací pomocí Swing frameworku, který je určen pro desktopové aplikace a je součástí standartní edice Javy.

Mezi další výhody tohoto frameworku patří integrace s dalšími frameworky jako je Hibernate, Spring apod., nebo možnost využití znovu použitelných komponent, což nám ušetří psaní stejných úseků kódu.

V době psaní této práce probíhá migrace naší aplikace z technologie JSP na technologii Wicket. Tato migrace obnáší převedení všech JSP stránek do validního HTML a propojení s logikou dané stránky[ET ()].

2.3.4 Framework Spring

Framework Spring je sada užitečných tříd a efektivních postupů pro vývoj aplikací. Zejména je používán pro tvorbu webových aplikací, ale stejně dobře lze využít na jakýkoliv typ aplikace. Cílem tohoto frameworku je zefektivnění, zjednodušení a zrychlení vývoje Java Enterprise aplikací. Spring zároveň nabízí možnost dobré testovatelnosti aplikace. Vzhledem k tomu, že lze jednotlivé objekty a vrstvy od sebe oddělit, můžeme jednoduše využívat jednotkových testů.

Tento framework, na rozdíl od jiných, podporuje konzistentním způsobem všechny vrstvy aplikace. Zároveň umožňuje integraci velkého množství prověřených nástrojů, a tudíž je možné jakoukoliv vrstvu nahradit jiným nástrojem, jako např. pro datovou vrstvu vybrat Hibernate. V aplikacích se Spring mimo jiné stará o bezpečné přihlašování, integraci se sociálními sítěmi a spoustu dalších funkcí, které jsou nutnou součástí každé moderní webové aplikace[de Velde(2007)].

2.3.5 Framework Spring MVC

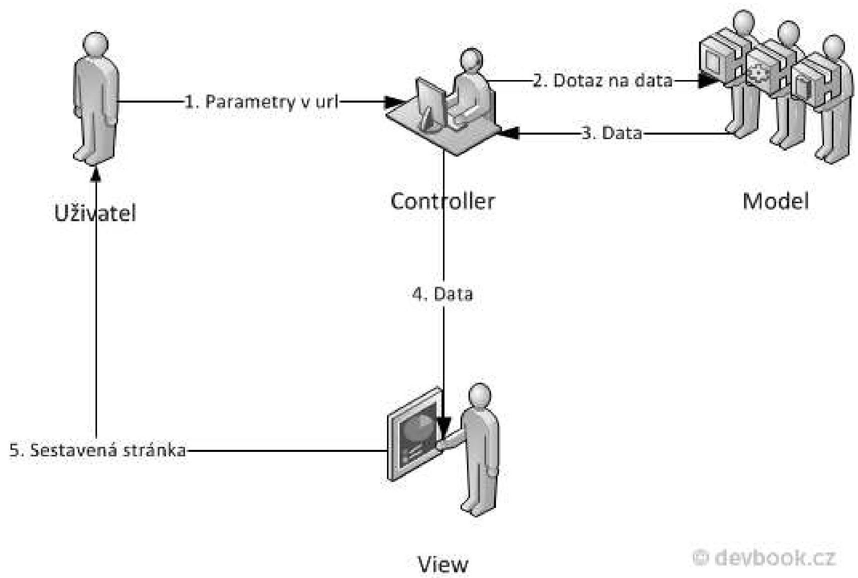
Tento framework je určen pro podporu webových aplikací. Spring MVC využívá návrhového vzoru Model-View-Controller. Tento návrhový vzor rozděluje prezenční vrtvu na tři části.

Model Doménové objekty nesoucí data. Na tyto data je Controllerem aplikován pohled.

View Uživatelem definovaný pohled. Jedná se tedy o pohled na nějaký konkrétní model vybraný na základě zpracování požadavku Controllerem.

Controller Zpracování požadavků, na jejichž základě vytvoří model, na který se poté aplikuje pohled.

Mezi velké výhody tohoto frameworku patří to, že je součástí aplikačního frameworku Spring, a tudíž může používat všechny jeho součásti. Další výhody jsou např. flexibilita, mapování hodnot z formulářů na objekty, podpora JSP a mnoho dalších[Kothagal(b)].



Obrázek 2.2: *Schéma fungování MVC*

2.4 Technologie v naší aplikaci

S pomocí definovaných technologií si nyní můžeme snadněji představit fungování naší aplikace. Naše aplikace je založena na principech třívrstvé architektury. Data jsou uložena a spravována v databázi Oracle.

Pro manipulaci s daty používáme framework Hibernate, kde pro mapování objektů je použito externích XML souborů. Mezi velkou výhodou frameworku Hibernate patří jeho spolupráce s frameworkem Spring, který je taktéž v naší aplikaci hojně používán. Prezenční vrstva naší aplikace využívá framework Spring MVC, díky čemuž je možné využívat propojení účtů uživatelů se sociálními sítěmi, zabezpečené přihlašování a spoustu nutných funkcí každé webové aplikace.

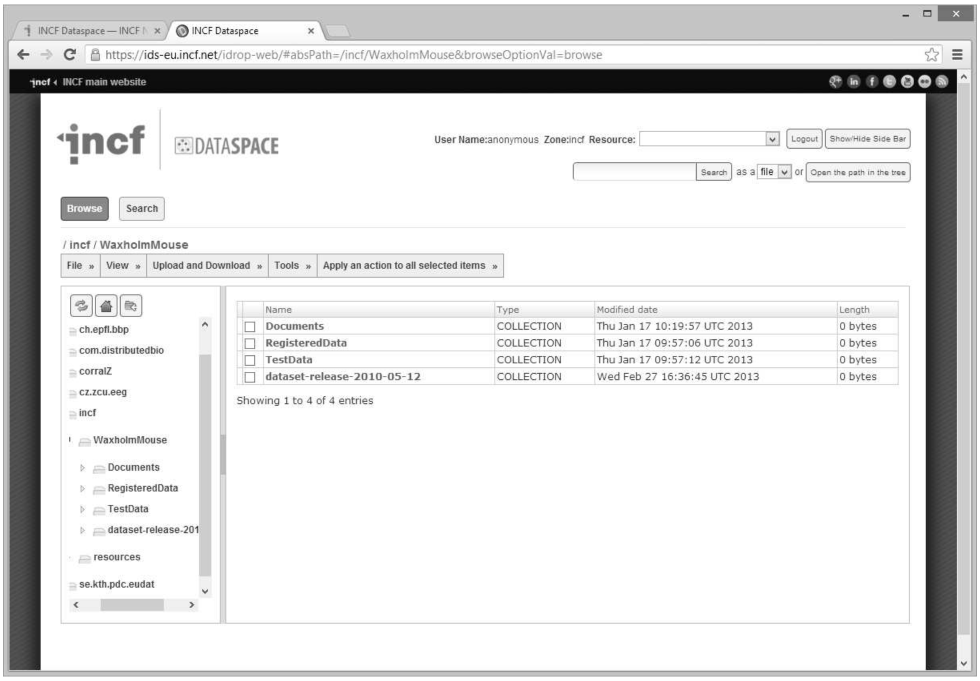
Uvedené použité technologie jsou platné v době psaní této práce. Vzhledem k tomu, že aplikace prochází neustálým vývojem a reaguje na nejnovější trendy v oblasti webových aplikací, je zřejmé, že i tyto technologie budou postupem času nahrazovány za efektivnější a modernější.

3 Přehled konkurenčních aplikací

V této části práce je úkolem analyzovat možnosti správy a zpracování EEG dat, která jsou nabízena konkurenčními aplikacemi. Na všechny aplikace je nahlíženo pohledem uživatele, jaké funkce daná aplikace nabízí. Cílem průzkumu bylo zjistit, zda-li neexistuje aplikace, která by se dala využít pro zpracování dat matematickými metodami, a to jak jednotlivé zpracování, tak i zpracování formou workflow.

3.1 INCF Dataspace

Tato aplikace je vyvíjena a spravována mezinárodní vědeckou organizací International Neuroinformatics Coordinating Facility (INCF), sídlící ve Švédsku. Hlavním úkolem této aplikace je umožnit uživatelům sdílení dat experimentů mezi ostatními uživateli. Přístup k datům je možný třemi způsoby. Jedním způsobem je přístup pomocí webové aplikace, další možnost je pomocí desktopové aplikace a poslední varianta je konzolové prostředí.



Obrázek 3.1: *Ukázka prostředí aplikace INCF Dataspace*

Sdílená data je možné popisovat metadaty. Mezi zajímavou funkci patří uspořádání grafických souborů do galerie.

Mezi další funkci patří vícenásobné stahování souborů, které funguje na principu nákupního košíku. Do košíku průběžně ukládáme data, a když je výběr kompletní, můžeme stáhnout vybraná data najednou[INC()].

Výhody:

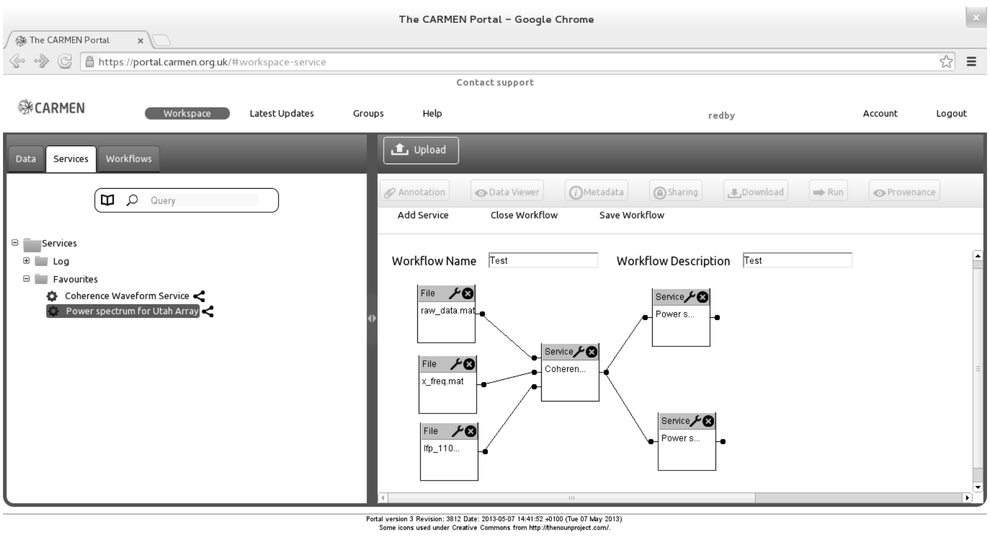
- Tři varianty přístupu k datům.

Nevýhody:

- Nepodporuje zpracování dat.

3.2 Portal Carmen

Tato aplikace je součástí projektu Carmen, který je financovaný britskou radou pro inženýrství a výzkum fyzikálních věd (Engineering and Physical Sciences Research Council). Hlavním úkolem aplikace je vytvoření virtuální laboratoře pro podporu e-Neuroscience. Jedná se o webovou aplikaci, což opět umožňuje téměř neomezenou dostupnost.



Obrázek 3.2: *Ukázka definice workflow v Portále Carmen*

Aplikace má intuitivní ovládání realizované pomocí záložek, které aplikaci rozdělují na tři části. Jedná se o části pro data, služby a workflow. V každé části je uživateli umožněno pracovat s konkrétními prvky, týkající se dané části.

Mezi hlavní funkce aplikace patří možnost ukládání dat a jejich sdílení s ostatními uživateli. Data lze opět popisovat metadaty. Aplikace nabízí možnost řízení přístupu k datům formou různých druhů oprávnění. Tyto druhy oprávnění lze použít i na metadata.

Další důležitou funkcí aplikace je možnost zpracování dat pomocí matematických metod. V současné době aplikace nabízí cca 100 různých metod, které si uživatel může zvolit pro zpracování. Aplikace podporuje metody napsané v jazycích Java, Python, Matlab a další. V případě zájmu uživatele o přidání další metody je nutné kontaktovat podporu aplikace. Každá metoda je popsána metadaty, která popisují např. vstupní a výstupní formát dat. Po nadefinování všech potřebných parametrů metody je daná metoda spuštěna na pozadí v nejbližším možném čase a výsledek je po zpracování uložen do databáze a zpřístupněn v aplikaci. V době psaní této práce byla aplikace rozšířena o podporu zpracování dat formou workflow. Definice workflow je realizována pomocí grafického nástroje, kde je celé workflow tvořeno diagramem na základě přidávání dat a metod uživatelem. Data jsou propojena s metodami tažením myši a celé workflow si může uživatel upravovat dle své představy. Jako vstupy do výpočetních metod lze použít samostatná data, či výsledky předchozích metod. Definované workflow je dále možné uložit pro pozdější úpravy či spustit zpracování definovaného workflow. Po ukončení celého workflow jsou výsledky uloženy a zpřístupněny uživateli.

Bohužel aplikace neumožňuje využívání poskytovaných služeb jinými způsoby, než pouze prostřednictvím samotné webové aplikace. Z tohoto důvodu nám není umožněno využít těchto služeb v naší aplikaci[Car()].

Výhody:

- Zpracování dat formou workflow.

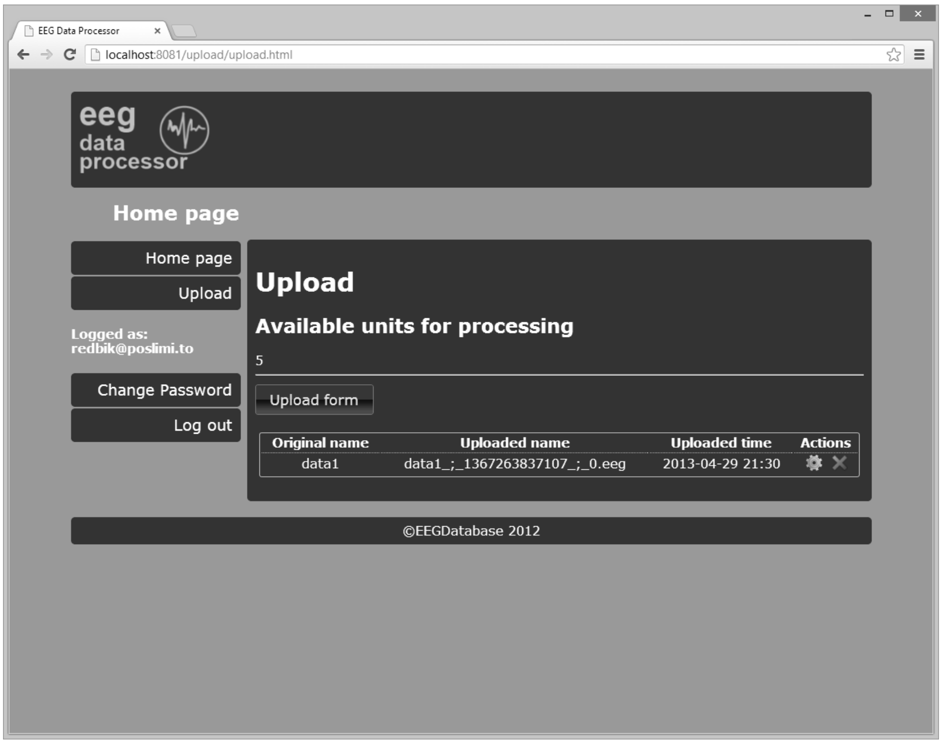
Nevýhody:

- Použití pouze jako webová aplikace.

3.3 EEG data processor

Tato aplikace je vyvíjena a provozována Katedrou informatiky a výpočetní techniky Západočeské Univerzity v Plzni. Oproti aplikaci EEGbase zde není možnost uložení dat, ale tato aplikace má na starosti pouze zpracovávání dat matematickými metodami.

Po nahrání dat, která chceme zpracovat, je k výběru několik matematických metod. Na základě zadaných parametrů metoda data zpracuje, a výsledek je uložen do dočasného uložiště. Uživatel si poté může vysledek zpracování stáhnout.



Obrázek 3.3: *Ukázka prostředí aplikace EEG data processor*

Aplikace podporuje přidávání dalších matematických metod vytvořených uživateli, které musí být napsány v jazyce Java, a splňovat definované požadavky dostupné v dokumentaci. Přístup k metodám je možný přes prostředí webové aplikace, nebo jako volání SOAP webové služby. V případě webové služby je výsledek vrácen jako pole bytů reprezentující xml volající aplikaci.

V současné době je možnost využít vstupní formát KIV FORMAT, který obsahuje vždy soubory EEG, VHDR a volitelně VMRK.

Výhody:

- Používání metod pomocí webové služby.

Nevýhody:

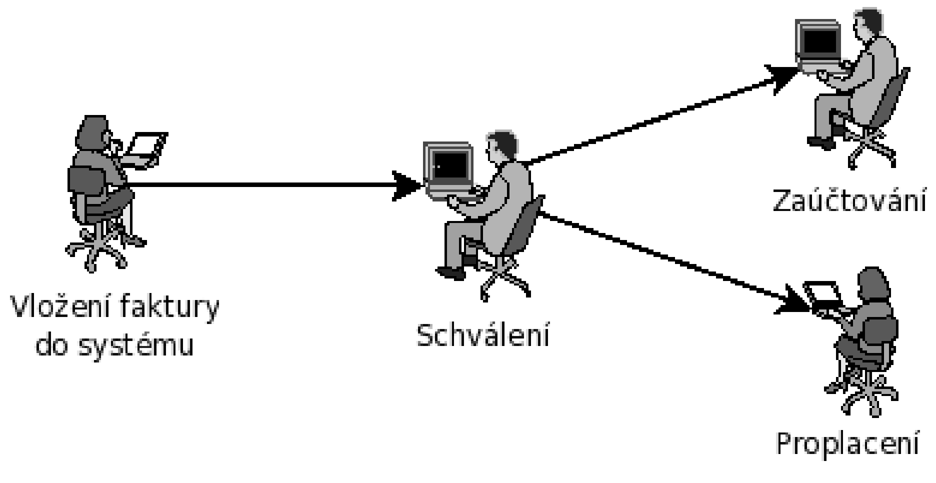
- Chybí podpora workflow.

4 Návrh systému workflow

V této části se blíže seznámíme s pojmem workflow a pokusíme se jej navrhnout společně s integrací do aplikace EEGbase. Součástí tohoto návrhu bude i nalezení poskytovatele zpracování dat matematickými metodami.

4.1 Workflow

Tímto termínem se rozumí postupné provádění komplexní činnosti, rozdělením na jednoduché činnosti a vazby mezi nimi. S workflow se můžeme setkat napříč různými odvětvími. Např. podnikové workflow může definovat zpracování přijatých dokumentů. Pomocí tohoto nástroje můžeme přesně definovat procesy od přijetí dokumentu až po jeho archivaci. Workflow definujeme tím, že proces zpracování dokumentu rozdělíme na jednotilivé kroky, určíme zúčastněné osoby a aktivity, které mají v každém kroku proběhnout.



Obrázek 4.1: *Schéma jednoduchého workflow*

Pomocí workflow je možná definice od jednoduchého procesu až po komplexní a rozsáhle procesy obsahující i několik desítek kroků a množství různých vazeb[Janišová(2012)].

4.2 Možnosti řešení

Systém workflow by měl umožnit uživatelům spouštět matematické metody nad daty formou sekvenčního zpracování. To znamená, že by mělo být umožněno uživatelům výběr většího množství dat, nad kterými budou spouštěny vybrané metody. Výsledky spuštěných metod by měly být uživatelům k dispozici až po kompletním dokončení všech zvolených metod. Uživatelé by měli mít možnost zvolit, zda následující metoda bude spuštěna nad jinými daty, či nad výsledky předchozích metod.

Na základě předchozího průzkumu konkurenčních aplikací jsou pro zpracování dat matematickými metodami vhodné pouze aplikace Portal Carmen a EEG data processor. Pouze jedna z těchto aplikací podporuje zpracování dat pomocí webových služeb. Touto aplikací je EEG data processor. Vzhledem k tomu, že se jedná o nutnou podmínku našeho návrhu, je pro návrh a následnou implementaci vybrána tato aplikace.

Při detailním pohledu na rozsah možného řešení se nám nabízejí dvě možnosti. První varianta spočívá v kompletní implementaci systému workflow v aplikaci EEGbase a EEG data processor využít jen pro jednotlivá volání. Jako druhá varianta je přenechat zpracování workflow aplikaci EEG data processor.

4.2.1 EEGbase řešení

Toto řešení spočívá v implementaci kompletního systému workflow pouze na straně aplikace EEGbase. EEGbase by měla na starosti kompletní zpracování workflow a webové služby by využívala pouze pro jednotlivé kroky workflow. Implemetaci by bylo možné rozdělit do následujích kroků.

Vytvoření klienta webové služby Jednalo by se o modul, který by byl schopen odesílat konkrétní požadavek webové službě a přijmout od ní výsledek. Pro každé zpracování by klient použil jedno volání.

Výběr dat a metod Implementace této funkce by měla zajistit výběr dat a metod požadovaných pro zpracování v systému workflow.

Integrace dat s klientem Vybraná data by byla postupně odesílána a přijaté výsledky shromažd'ovány. Po dokončení celého workflow by byly zpřístupněny uživatelům.

Výhody:

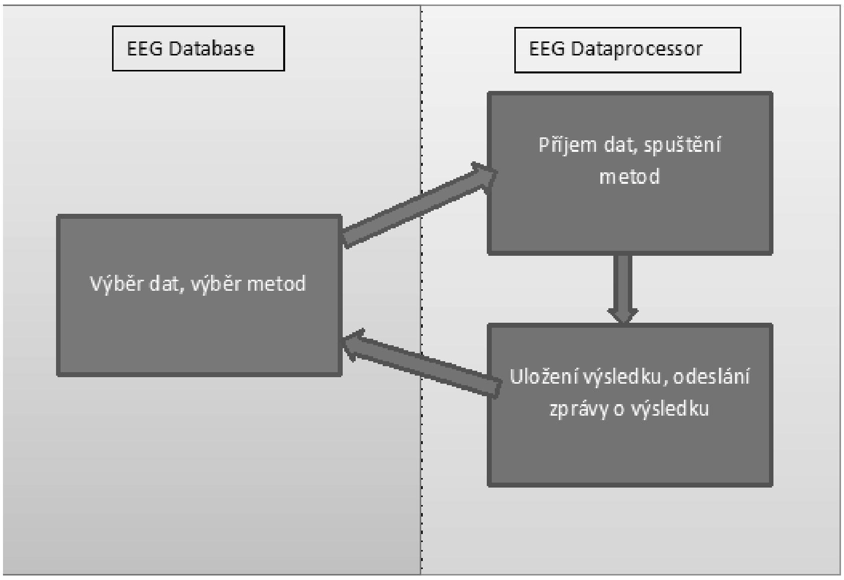
Veškerá správa workflow by byla úkolem aplikace EEGbase a nebyl by nutný zásah do aplikace EEG data processor.

Nevýhody:

Značně zvýšená zátěž serveru při zpracování metod a při neustálé komunikaci s webovou službou.

4.2.2 EEGbase a EEG data processor

Jedná se o řešení, které zahrnuje úpravu obou aplikací. EEGbase by vystupovala pouze v roli klienta. Popis workflow spolu s daty by odeslala webové službě, která by se postarala o zpracování a vrátila by kompletní výsledek. Opět je možné implementaci rozdělit do následujícíh kroků.



Obrázek 4.2: *Jednoduché schéma řešení*

Úprava EEG data processoru Webová služba by musela být upravena tak, aby byla schopna přijmout více dat spolu s popisem procesů, kterými se data mají zpracovávat. Zároveň by úprava musela obsahovat vrácení všech výsledků z provedeného zpracování.

Úprava EEGbase Po výběru konkrétních dat a metod by všechna data byla odeslána spolu s popisem workflow webové službě a po dokončení by byly vráceny konkrétní výsledky.

Výhody:

Přenechání kompletní správy workflow EEG data processoru, čímž nedojde k zatížení naší aplikace.

Nevýhody:

Nutnost úpravy obou aplikací, což s sebou nese větší časové nároky na realizaci.

4.3 Výběr řešení

Vzhledem k možnostem, které jsou nám nabízeny druhou variantou, jsem se rozhodl pro implementaci tohoto řešení. Tato varianta nám nabízí možnost ponechání aplikace EEGbase jako samostatná data, spravující aplikace s možností vzdáleného zpracování dat. Další bezespornou výhodou je rozšíření aplikace EEG data processor o podporu workflow, čehož mohou využít i ostatní uživatelé této aplikace.

4.4 Detailní návrh workflow

Cílem návrhu je nabídnout efektivní řešení zpracování dat, pomocí uživateli definovaného workflow. Ke zpracování dat bude použita webová služba aplikace EEG data processor.

Klient této webové služby, který bude požadovat workflow zpracování, společně s daty, odešle i popisný soubor, který bude definovat metody a data, která si přeje zpracovat v jednotlivých krocích.

Celé workflow bude rozděleno do jednotlivých celků, označených jako "workunit". Tyto jednotky lze použít na logické a přehledné uspořádání více experimentů v rámci jednoho workflow. Každý tento celek bude rozdělen na daný počet kroků "workstep", které budou obsahovat požadavanou metodu s parametry, název konkrétních dat a další parametry.

U každého kroku si uživatel může zvolit, zda-li si přeje zpracovávat původní data, či výsledek některého z předchozích kroků. V tomto případě je nutné zajistit, aby první krok vždy obsahoval původní data.

V aplikaci EEG data processor je striktně zadán požadavek na vstupní data každé metody. Je tedy nutné zajistit, aby v případě použití výsledku byl daný výsledek požadovaného formátu. Z tohoto důvodu bude při každém vstupu provedena kontrola formátu dat, protože různé metody vracejí různé výsledky.

Mezi další parametry každého kroku bude patřit informace, zda-li je požadován konkrétní výsledek zaslat volajícímu, či bude pouze použit v jiném kroku obsaženém ve workflow. Posledním parametrem kroku bude formát vstupních dat, pro identifikaci, zda se jedná o data původní či výsledná.

Díky tomuto jednoduchému a jednoznačnému popisu workflow jsme schopni realizovat následující uživatelské scénáře.

Jednoduché zpracování Jedná se o základní stavební kámen celého systému. Požadavkem je zpracování původních dat vybranou metodou. V tomto případě je možné zvolit, zda-li si přejeme získat výsledek, či ho jen dočasně uložit pro pozdější použití.

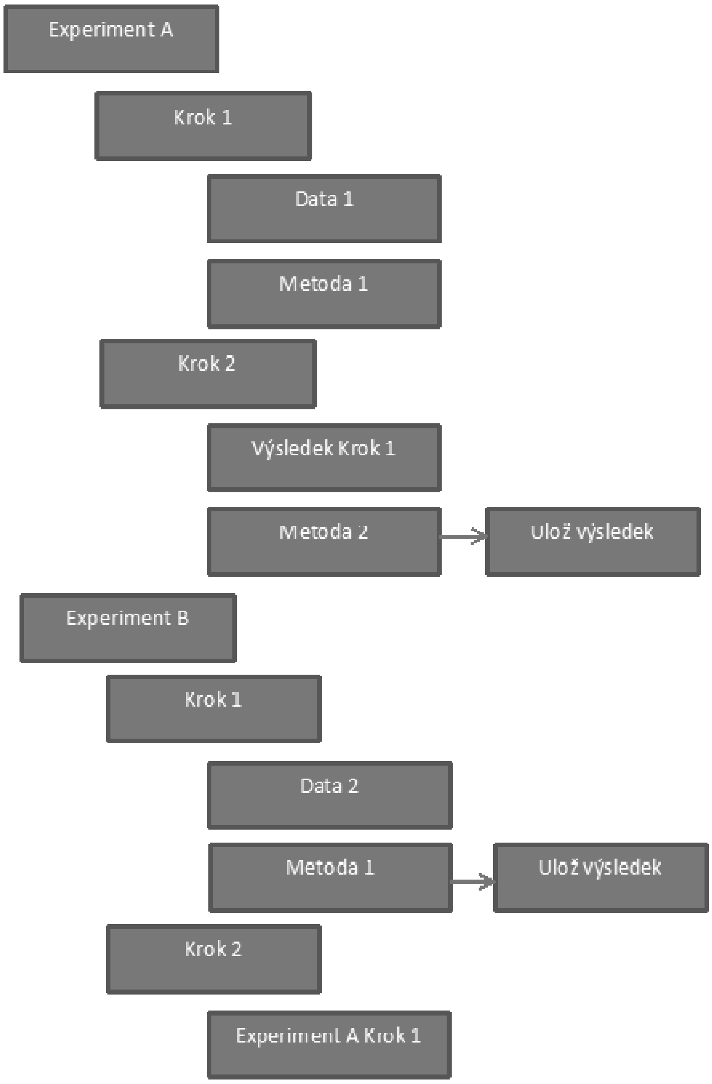
Násobné zpracování Je možné vygenerovat popisný soubor, který na základě informací od uživatele vygeneruje daný počet kroků. V tomto případě je potřeba zajistit, aby vybraná metoda vracela požadovaný formát dat, který je vhodný pro opakované zpracování.

Násobné metody Ná základě informací od uživatele je možné vygenerovat daný počet kroků se stejnými daty, ale rozdílnými metodami.

Tyto uživatelské scénáře jsou v plné kompetenci klienta webové služby, který má na starosti dle informací od uživatele generovat popisný soubor s patřičnými kroky.

Výsledky daných kroků budou ukládány pod jménem složeného z názvu "workunit"a "workstep". Díky tomuto pojmenování bude možné přistoupit k jakémukoliv výsledku v rámci celého workflow.

Po zpracování všech kroků bude sestaven výsledek z požadovaných výsledných dat, který bude poté vrácen jako návratová hodnota volajícímu.



...

Obrázek 4.3: *Ukázka možného workflow*

4.5 Detailní návrh klienta

Vzhledem k tomu, že se jedná o velmi komplexní a rozsáhlou problematiku zpracování workflow, bude klient sloužit jako funkční prototyp zejména k otestování funkčnosti navrženého systému. Tento klient bude integrován do aplikace EEGbase, kde bude plnit zejména úlohu získání dat od uživatele.

Po získání uživatelských dat prostřednictvím formulářů, bude sestaven seznam vstupních dat, která budou dle názvu a ID experimentu získána přímo z databázového uložiště. Na základě informací popisující workflow, bude vygenerován popisný soubor, který bude spolu s daty odeslán webové službě. Proces volání služby bude spuštěn na pozadí, aby uživatelům nebránil v další práci s aplikací. Po získání výsledků, budou tyto výsledky uloženy do databáze a přístupné uživatelům z prostředí aplikace.

Klient bude dále rozvíjen do plně funkčního řešení výzkumnou skupinou, zabývající se vývojem a provozem aplikace EEGbase.

5 Implementace a integrace workflow

V této části implementujeme zpracování workflow do webové služby aplikace EEG data processor a dále tuto funkcionalitu integrujeme do aplikace EEGbase.

5.1 Implementace workflow

V této části se věnujeme popisu implementace na straně aplikace EEG data processor.

Zpracování workflow úkolů je implementováno jako rozšíření současné webové služby aplikace EEG data processor. Webová služba umožňuje získání seznamu dostupných metod, získání seznamu parametrů dané metody, získání počtu aktuálně volné kapacity pro zpracování dat a zpracování dat matematickou metodou. Aplikace podporuje vstupní formát EEG souborů a vrácení výsledku ve formě xml.

Po rozboru požadovaných funkcí z našeho návrhu můžeme zpracování všech kroků rozdělit do dvou skupin.

Zpracování EEG souborů Tato funkcionalita je již v aplikaci implementována. Jedná se o zpracování souborů formátu EEG požadovanou metodou a vrácení výsledku ve formátu xml.

Zpracování výsledků Jedná se o funkci, kdy vstupem pro požadovanou metodu je výsledek některého z předchozích zpracování. V tomto případě je potřeba zajistit, aby daný vysledek byl ve formátu podporující matematické metody, což je v našem případě pole typu double, které nejlépe reprezentuje naměřené hodnoty EEG.

Celé workflow bude rozděleno do samostatných kroků, ve kterých bude definováno následující:

Vstupní data Jedná se o název konkrétních dat. V našem případě tedy název vstupních EEG souborů, případně název kroku, z kterého chceme použít výsledek.

Formát vstupních dat Současná aplikace EEG data processor podporuje pouze formát EEG, který je definován hodnotou KIV_FORMAT. V rámci naší implementace bude aplikace rozšířena o DOUBLE_FORMAT, který bude definovat data získaná z předchozích zpracování.

Uchování dat Volba uživatele, zda si data získaná spuštěním dané metody přeje zaslat mezi výsledky, či tato data budou použita jako vstup pro některé z následujících zpracování.

Název metody Název požadované metody.

Parametry metody Vstupní parametry vybrané metody.

Tímto jednoduchým a jednoznačným popisem každého kroku bude možné definovat téměř jakékoliv workflow, které bude popsáno popisným souborem. Uživatelské scénáře typu vícenásobného spuštění metod budou realizovány daným počtem kroků dle počtu požadovaných spuštění.

Webová služba na vstupu získá všechny EEG soubory, které se budou ve workflow používat a spolu s nimi i popisný soubor, který bude definovat celé workflow.

Na základě tohoto popisného souboru bude vytvořen seznam kroků obsažených v celém workflow. Každý krok bude poté spuštěn a zpracováván dle uvedených parametrů.

5.1.1 Popisný soubor

Jako formát popisného souboru bylo zvoleno XML, hlavně z důvodu rozšířenosti tohoto formátu a snadné implementaci pomocí již existujících nástrojů od třetích stran.

Kořenovým elementem celého dokumentu je "workflow". Po přijetí popisného souboru je zkontrolováno, zda-li je obsažen kořenový element, tedy že se jedná o popis workflow. V případě, že element není nalezen, je vyhozena vyjímka a vrácena volajícímu. Element "workflow"obsahuje pouze jeden parametr, a tím je jméno.

Workflow obsahuje elementy "workunit", které reprezentují logický celek a slouží hlavně k přehlednějšímu popisu a možnosti rozdělit workflow do skupin, kde každá skupina obsahuje určitý počet kroků. Tento element má za parametr pouze jméno, které slouží k vytvoření jména výsledku, což dovoluje k výsledkům přistupovat odkudkoliv z celého workflow.

Element "workunit"obsahuje další elementy "workstep", které reprezentují daný krok obsahující již parametry týkající se konkrétního spouštění metod. Mezi parametry patří jméno, které slouží také k pojmenování výsledku. Jméno výsledku je vždy složeno z jména "workunit"a "workstep". Dalším parametrem je formát vstupních dat, který může nabývat hodnot KIV_FORMAT a DOUBLE_FORMAT. Dle tohoto parametru je možné krok rozdělit do jedné ze dvou skupin potřebných pro spouštění. Posledním parametrem je "store", který nabývá logických hodnot a informuje o tom, zda-li si uživatel přeje výsledek uchovat či nikoliv.

Každý "workstep"obsahuje dva elementy, kterými jsou názvy dat pro zpracování a název metody spolu s jejími parametry.

Listing 5.1: Ukázka popisného souboru

```
<?xml version="1.0" encoding="UTF-8"?>
<workflow name="Workflow">
  <workunit name="Experiment1">
    <workstep name="SimpleFile" format="KIV_FORMAT" store="
       false">
      <data>data1.eeg</data>
      <data>data1.vhdr</data>
      <method params="01,100,Cz,FAST_DAUBECHIES_2">DWTPlugin
         -1.0.0</method>
    </workstep>
    <workstep name="SimpleDouble" format="DOUBLE_FORMAT" store
       ="true">
      <data>Experiment1_SimpleFile</data>
      <method params="01,1000,Cz,COMPLEX_GAUSSIAN
         ,1,1,1,14000,14000">CWTPlugin-1.0.0</method>
    </workstep>
  </workunit>
  <workunit name="Experiment2">
    <workstep name="SimpleFile" format="KIV_FORMAT" store="
       true">
      <data>data2.eeg</data>
      <data>data2.vhdr</data>
      <method params="01,100,Cz,FAST_DAUBECHIES_2">DWTPlugin
         -1.0.0</method>
    </workstep>
  </workunit>
</workflow>
```

5.1.2 Chybové stavy

Aplikace bude rozšířena o zpracování chybových stavů pomocí návratových kódů. I přes aparát vyjímek, které Java nabízí, byl tento způsob zvolen z důvodu, že tyto kódy jsou vráceny spolu s výsledky. Chybové kódy umožňují klientovi vlastní definici chybových zpráv, což lze ocenit při lokalizaci a možnost implementace specifického chování klienta na daný kód.

V tuto chvíli je aktivní podpora těchto dvou kódů.

1. Nepodporovaný formát vstupních dat.
2. Použitý výsledek není formátu DOUBLE ARRAY.

Do budoucna je možné rozšíření o další kódy, případně sjednocení chování celé aplikace při chybových situacích.

5.1.3 Implementace metody pro zpracování workflow

Webová služba bude rozšířena o novou metodu, která bude používána výhradně na volání processorů pro každý krok. Tato metoda jako parametr získá všechny potřebné EEG soubory spolu s popisným souborem. Po ukončení metody bude vráceno pole bytů, které bude obsahovat všechny výsledky.

Na začátku tato metoda vytvoří objekt "DataHandler", který má na starosti manipulaci se vstupními daty. Dále je vytvořen objekt "ParameterHandler", který má na starosti vytvoření seznamu kroků, včetně všech parametrů obsažených v popisném souboru.

V dalším kroku je od objektu "ParameterHandler"získán seznam všech kroků, který je postupně zpracováván požadovanými matematickými metodami.

Pro každý krok je volán jeden ze dvou processorů podle formátu vstupních dat. Výsledná data jsou poté objektem "DataHandler"uložena do seznamu výsledků.

Po zpracování všech kroků je od "DataHandleru"získán finální výsledek, sestavený ze seznamu výsledků ná základě parametru o uložení. Tento finální výsledek je vrácen volajícímu v podobě pole bytů.

Tato metoda dále zajišťuje nahrazení výsledku chybovým kódem, v případě chyby vstupních dat.

5.1.4 Implementace zpracování popisného souboru

Zpracování popisného souboru zajišťuje objekt třídy "ParameterHandler". Tento objekt má dvě proměnné. Jednou z nich je seznam všech kroků workflow, získaný z popisného souboru. Druhou proměnnou je řetězec reprezentující popisný soubor.

Tento řetězec je předán konstruktoru "ParameterHandleru", ve kterém je zároveň naplněn seznam kroků, parsováním předaného řetězce. Pro parsování je vybrána metoda DOM, která umožňuje v případě velkých XML souborů, menší paměťovou náročnost, protože se parsuje postupně a nejsou z každého elementu vytvářeny objekty. Další důvod byla vlastní zkušenost s touto metodou.

Na základě postupného parsování jsou vytvářeny objekty "Workstep", které obsahují pouze proměnné reprezentující hodnoty a parametry každého kroku. Každý takto vytvořený objekt je uložen do seznamu kroků, který je poté používán v metodě webové služby.

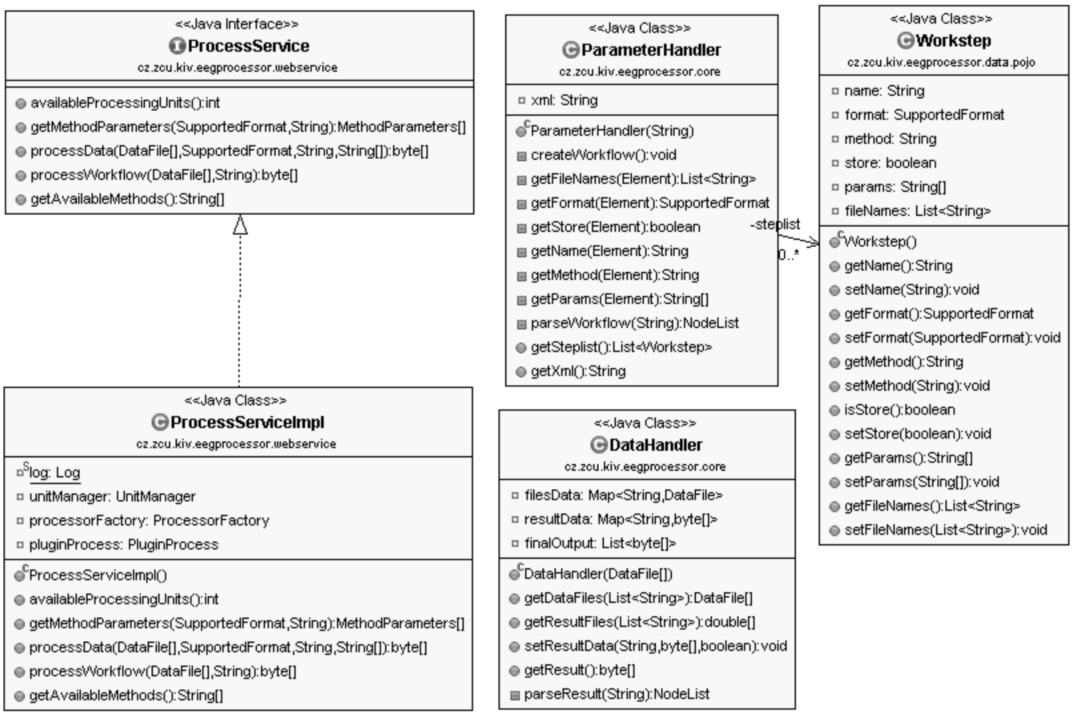
5.1.5 Implementace zpracování dat

Data jsou zpracovávány pomocí objektu "DataHandler". Tato třída má tři hlavní proměnné. Jedná se o seznam vstupních EEG souborů, seznam výsledků spuštěných metod a seznam finálních výsledků, které jsou požadovány od volajícího. Seznamy vstupních dat jsou předány pomocí rozhraní Map, čímž je zajištěno rychlého přístupu ke konkrétním datům. Klíčem je vždy buď název souboru, nebo kroku, ze kterého je získán výsledek.

Při vytváření objektu "DataHandler"je v konstruktoru předáno pole vstupních EEG souborů, kterým je naplněn seznam vstupních souborů, a přiřazení klíčů, dle názvů souborů. Seznam výsledků je naplňován vždy po zpracování konkrétního kroku a získání výsledku.

Objekt "DataHandler"zajišťuje pro webovou službu předávání požadovaných vstupních dat, ukládání výsledků spuštěných metod, převod výsledků do formátu pole double, kontrola výsledků, zda jsou formátu pole double a zajištění vrácení požadovaných výsledků ve formátu pole byte.

Požadované výsledky jsou vždy formátu XML, který je převeden do pole bytů a poté jsou serializací všechny výsledky převedeny do jednoho pole bytů, které je vráceno klientovi. Úkolem klienta je poté deserializací z tohoto pole získat opět pole řetězců reprezetující dané xml výsledky.



Obrázek 5.1: *Class diagram úprav*

5.1.6 Možné vylepšení v rámci workflow

Aplikaci je dále možné rozšířit o zpracování více druhů chybových stavů. Dále se zde nabízí možnost integrace zpracování workflow do prostředí webové aplikace, kde by bylo uživatelům umožněno definovat a zpracovat workflow jen za pomocí samotné aplikace EEG data processor.

5.2 Implementace klienta

V této části se věnujeme popisu implementace a integrace klienta webové služby zpracovávající workflow do aplikace EEGbase. Integrace klienta je implementována jako soubor funkcí zajišťující následující služby.

Získání dat od uživatele Uživatel má možnost přímo z webového prostředí aplikace zadat všechny potřebné vstupy pro zpracování vybraných dat, včetně parametrů každého kroku.

Vytvoření popisného souboru Na základě získaných vstupů je vytvořen popisný soubor pro workflow, který je zasílán webové službě spolu s datovými soubory.

Získání potřebných datových souborů Dle uživatelem vybraných experimentů jsou z databáze získány potřebné soubory, které budou zpracovány.

Komunikace s webovou službou Zajištění komunikace s webovou službou za účelem zpracování workflow, získání seznamu aktuálně podporovaných metod a získání výsledků zpracování.

Uložení výsledků workflow Uložení výsledků zpracování dat do databáze, aby uživatel poté k nim mohl přistupovat z prostředí webové aplikace.

Klient je implementován v podobě funkčního prototypu, jehož hlavním cílem je otestování funkčnosti navrhnutého systému workflow. Tento klient bude dále vyvíjen do plně funkčního řešení, integrovaného do aplikace EEGbase, respektující současný vzhled, chování a funkce aplikace.

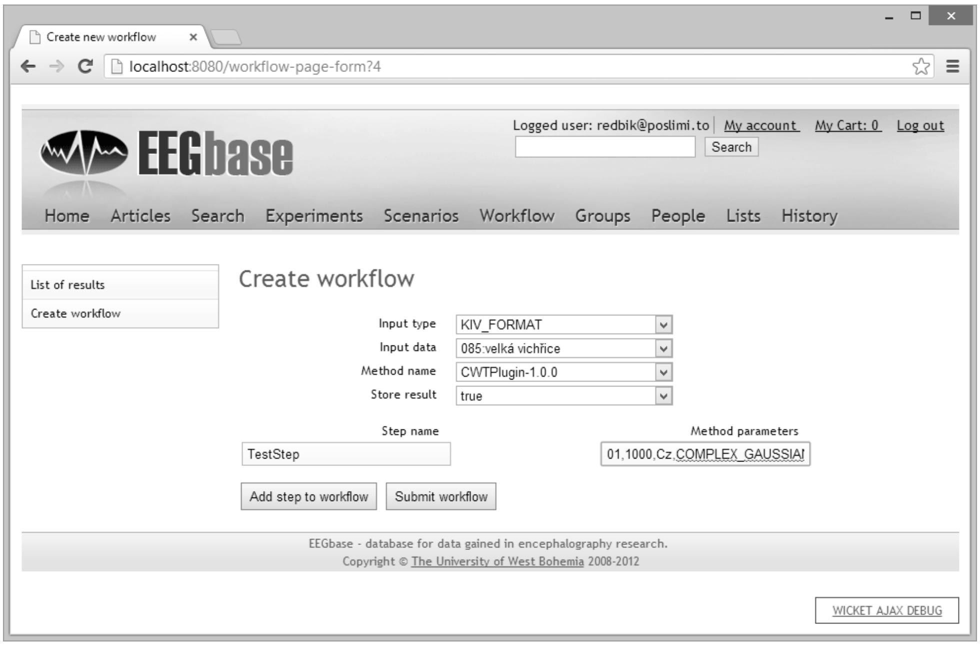
Uživatelské prostředí klienta je realizováno pomocí wicket frameworku, na který v současné době probíhá migrace celé aplikace.

5.2.1 Implementace získání vstupů

Současná aplikace je rozšířena o nové stránky ”WorkflowFormPage”a ”WorkflowListResultPage”. Tyto stránky jsou integrovány do hlavního menu pod nově přidanou záložku ”Workflow”. Obě stránky obsahují menu s výběrem konkrétní stránky umístěné na levé straně. Jako defaultní stránka je zvolena stránka zobrazující výsledky zpracovaných dat, která bude v rámci přechodu na technologii wicket rozšířena do plně funkční podoby.

Druhá stránka obsahuje potřebné prvky pro získání vstupů od uživatele. Pro vstupy jako jsou formát vstupních dat, data vstupu, výběr metody a volba uložení výsledku je zvolena wicket komponenta "DropDownChoice". Pro název přidávaného kroku a parametry metody je zvolena komponenta editovatelného "TextFieldu". Posledními ovládacími prvky jsou komponenty "AjaxButton", které zajišťují přidání kroku a spuštění definovaného workflow. Pro informace o zvolené funkci je k dispozici spodní část pod ovládacími prvky, kde se vždy zobrazí text k zvolené funkci, jako např. informace o přidání kroku do workflow.

Vzhledem k volbě mezi dvěma možnými vstupy dat, je implementována provázanost mezi "DropDownChoice", kde na základě vybraného typu je druhý prvek naplněn požadovaným seznamem vstupních dat. Tato funkce byla získána pomocí přidání komponenty "AjaxFormComponentUpdatingBehavior", která reaguje na změnu dat při běhu aplikace. K dispozici jsou dva seznamy, kde jeden obsahuje seznam všech experimentů včetně ID a druhý názvy všech předchozích kroků.



Obrázek 5.2: *Formulář pro vstup uživatele*

Seznam všech experimentů a dostupných metod je získán při vytvoření stránky ”WorkflowFormPage”. Seznam kroků je aktualizován vždy při přidání každého kroku.

Název kroku a seznam parametrů je zadán do vstupních polí, která jsou vždy po přidání kroku vymazána. Před přidáním kroku je dle vybraného experimentu stáhnut z databáze seznam ”fileId”všech data souborů, ale pouze v případě, že se jedná o soubory formátu EEG,VHDR a VMRK, které jsou podporovány webovou službou. K experimentům v databázi je přistupováno pomocí Spring beany ”ExperimentFacade”, která má na starosti manipulaci s experimenty v databázi. Následně všechny získané vstupy jsou předány objektu ”WorkflowService”, který je dále zpracovává.

5.2.2 Implementace zpracování vstupů

Zpracování získaných vstupů zajišťuje objekt třídy ”WorkflowService”, který je vytvořen jako beana pomocí frameworku Spring při vytvoření stránky ”WorkflowFormPage”. Tento objekt obsahuje metody pro vytváření seznamu kroků, generování popisného souboru, získávání seznamu dostupných metod a spouštění zpracování workflow.

V současné době je k dispozici možnost generovat popisný soubor v rámci jedné ”workunit”, s možností jednoduchého rozšíření klienta o více těchto jednotek. Po získání všech vstupů je zavolána metoda ”addToWorkflow”, která z předaných vstupů vytvoří část popisného souboru odpovídající jednomu kroku. Tato část, reprezentována vytvořeným řetězcem, je vložena do seznamu kroků. Tento seznam je využíván při volbě vstupů a při pojmenování výsledků.

Vzhledem k jednoduchosti popisného souboru je definice kroku generována pouze doplněním předaných hodnot do připravených řetězců, které jsou poté spojeny do jednoho celku. V porovnání s použitím nástrojů DOM či JAXB se tento přístup zdá jako optimální.

Po vytvoření všech kroků je volána metoda ”runService”, která vytváří ze seznamu kroků kompletní popisný soubor, a poté nové vlákno, ve kterém je workflow dále zpracováváno. Vlákno je v samostatné třídě, která v konstruktoru přijímá objekt třídy ”PersonDao”, vytvořený opět jako Spring beana. Díku tomuto objektu je možné získat aktuálně přihlašeného uživatele, jehož data jsou použita při ukládání výsledků do databáze.

V nově vytvořeném vláknu je nejdříve vytvořen seznam objektů "ServiceResult", v počtu odpovídajícím uživatelem požadovaných výsledků. Tyto objekty jsou poté uloženy do databáze pomocí Spring beany "ServiceResultDao", se statusem running. Toto nám umožňuje získat přístup k seznamu běžících metod a očekávaných výsledků již v okamžiku spuštění worklow.

Následně jsou z databáze stáhnuty požadované datové soubory podle získaného seznamu "fileIds"a Spring beany "FileFacade". Tyto soubory jsou přidány do seznamu vstupních souborů, který je požadován webovou službou jako jeden ze vstupů.

Předposledním krokem nového vlákna je vytvoření objektu třídy "DataProcessor", který zajišťuje konkrétní volání webové služby. Výsledek vrácený voláním je předán metodě "storeResult", která zajišťuje aktualizování databáze o získané výsledky. K aktualizaci databáze je využit dříve vytvořený seznam "ServiceResultů", kde každý výsledek aktualizujeme na stav finished a připojíme výsledná data ve formátu Blob.

Mezi další metody třídy "WorkflowService", patří metoda pro získání seznamu všech experimentů z databáze, ze kterých se vytváří název obsahující popis a ID experimentu. Takto vytvořený seznam je používán při výběru dat uživatelem. Dále je zde metoda pro získání seznamu aktuálně podporovaných metod webovou službou, která k tomu využívá volání webové služby. Další důležitou metodou je vytváření a aktualizování seznamu "fileId", ke kterému je použito rozhraní Set. Toto rozhraní zajišťuje, že každé "fileId"je v seznamu uloženo pouze jednou i v případě, že uživatel zvolil stejný experiment v různých krocích.

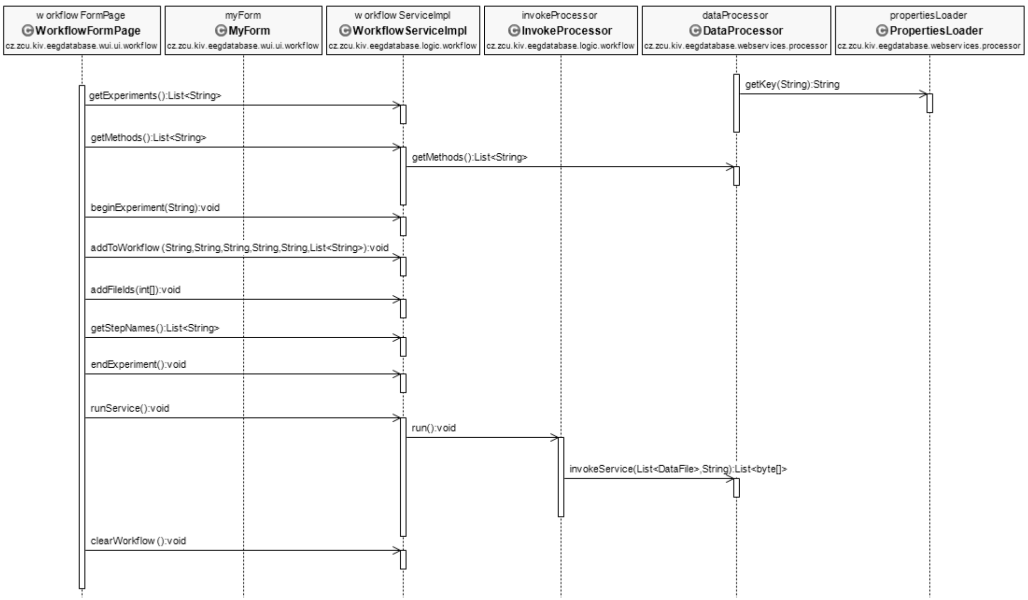
5.2.3 Implementace komunikace s webovou službou

Komunikaci s webovou službou zajišťuje objekt třídy "DataProcessor", který v současné implementaci disponuje metodou pro zpracování workflow a získání seznamu aktuálně podporovaných metod. Při vytvoření objektu je pomocí třídy "PropertiesLoader"načten soubor setting.properties, který obsahuje adresu koncového bodu webové služby, uživatelské jméno a heslo pro přístup k aplikaci EEG data processor.

Třídy reprezentující objekty webové služby jsou vygenerovány pomocí aplikace Apache CXF, která je používána v implementaci webové služby.

Metoda pro získání seznamu podporovaných matematických metod nejdříve vytvoří službu a potom vzdáleným voláním získá seznam podporovaných metod.

Metoda pro zpracování workflow přebírá mezi parametry seznam datových souborů a popisný soubor. Tyto dva parametry jsou poté předány vzdálené metodě. Jako výsledek je získáno pole bytů, které deserializací je převedeno do seznamu polí bytů. Každý prvek tohoto seznamu reprezentuje jeden výsledek. Vzhledem k různé délce času zpracování dat, je čekání na výsledek realizováno při otevřeném spojení.



Obrázek 5.3: *Sekvenční diagram klienta*

5.2.4 Možné vylepšení klienta

Klienta je možné rozšířit o podporu více workunit, kterými je umožněno lepší organizace workflow. Dále je třeba rozšíření o manipulaci s výsledky, která v současné implementaci není zahrnuta. Vzhledem k tomu, že se jedná o funkční prototyp, je zde mnoho možných a zároveň i potřebných vylepšení. Do budoucna se nabízí možnost změny klienta na grafický nástroj, kde si uživatel bude moci definovat workflow pomocí vizualizačních nástrojů.

5.3 Testování

V první fázi vývoje byla aplikace testována pomocí konzolového klienta, který byl navržen pouze za účelem testování. Data, která byla použita v těchto testech, byla získána z databáze aplikace EEGbase, a byl k nim napsán konkrétní popisný soubor. V tomto souboru byly definovány tři kroky. V prvním kroku se jednalo o vstup původních dat, v dalším kroku byl pro vstup metody zvolen výsledek předchozí metody a poslední krok testoval možnost rozdělení workflow na více celků pomocí "workunit". Zároveň byla i testována funkcionalita vrácení výsledků, kdy byly požadovány pouze dva výsledky ze třech vstupů.

V další fázi vývoje byl testován klient integrovaný do aplikace EEGbase. Nejdříve bylo testováno jádro klienta za pomocí pevně stanovených parametrů v programu. Uživatelský formulář byl testován v poslední fázi pomocí definování vlastního workflow o několika krocích a kontrola uložených výsledků v databázi.

Dálší testovací scénáře by měli být realizovány v rámci dalšího vývoje systému workflow a jeho integrace. Je nutné aplikovat i regresní testy pro obě části systému a to vzhledem k tomu, že implementace v několika případech zasahuje do jádra aplikací.

6 Závěr

Vzhledem ke stále se zdokonalujícím technologiím je pro spoustu uživatelů aplikací důležité to, jak jim práce se samotnou aplikací dokáže ušetřit jejich čas. Ve většině odvětví se velmi často skloňuje termín automatizace, který značí nalezení cesty, jak obvyklé manuální činosti lze dělat automaticky. Jedním takovým nástrojem, který uživatelům šetří čas a zároveň něco automatizuje, je právě workflow, o kterém je tato práce.

Úkolem práce bylo navrhnout a implementovat systém, který uživatelům umožní definování a používání workflow při zpracování dat EEG/ERP.

Vzhledem k tomu, že se jedná o velmi rozsáhlou oblast, je řešení ve formě funkčního prototypu, který je připraven na další rozšiřování. Oblast workflow je stále rychle se vyvíjející oblast, kde je velmi mnoho prostoru na zdokonalení a vylepšení. Současný model je schopen představit možnosti použití workflow systému v oblasti EEG/ERP.

Tato práce pro mne byla velmi přínosnou, protože mě částečně zasvětila do vývoje webových aplikací, se kterými nemám žadnou zkušenost z minulosti, a také mi ukázala cestu, jakým způsobem lze naše aplikace stále zdokonalovat.

A Uživatelská dokumentace

V této příloze je popis nutných kroků k realizaci workflow z prostředí aplikace EEGbase.

Po přihlášení do aplikace se funkce workflow nachází v horním záložkovém menu pod popisem Worklow. Pod touto záložkou je defaultně k dispozici seznam uložených výsledků v databázi. V levé části je umístěno menu na výběr mezi zobrazeným seznamem či vytvořením nového workflow.

Po vybrání položky Create worklow se dostaneme na vstupní formulář, ve kterém definujeme workflow v následujících krocích

1. Výběr formátu vstupních dat.
2. Výběr dat, která si přejeme zpracovat.
3. Výběr požadované metody.
4. Volba, zdali si přejeme výsledek uchovat.
5. Název právě definovaného kroku.
6. Vstupní parametry dané metody.
7. Uložení kroku pomocí tlačítka Add to workflow.
8. Při posledním kroku potvrzení tlačítkem Submit.

Výběr formátu vstupních dat je možný mezi formátem KIV_FORMAT a DOUBLE_FORMAT. Pokud zvolíme první volbu, jsou nám k dispozici všechny experimenty, z kterých si můžeme vybrat v poli Input data. V případě volby druhého formátu jsou k dispozici již uložené kroky. Z tohoto důvodu je tato volba při definici prvního kroku prázdná.

Výběr metod probíhá na základě seznamu získaných podporovaných metod. K dispozici je vždy aktuální seznam metod.

Volba Store result slouží k volbě, zda-li si uživatel přeje výsledek definovaného kroku zaslat zpět, či ho bude pouze používat v dalších krocích.

V dalších polích je vyplněno jméno kroku, které je využíváno k přístupu k výsledkům a pojmenování ukládáných výsledků do databáze.

Poslední vstupním polem je část pro zadání parametrů dané metody. Parametry se oddělují pomocí čárky. Při obouch typech vstupu je nutné definovat všechny parametry s tím, že v případě použití výsledků jsou první tři parametry ignorovány, protože se týkají pouze datových souborů.

Při vyplnění všech polí je k dispozici tlačítko Add to workflow, který námi definovaný krok přidá a všechny hodnoty formuláře se nastaví na výchozí hodnoty, čímž je možné definovat další krok. V případě, že se jedná o poslední krok workflow, je k dispozici tlačítko Submit, který spustí celý proces workflow a aplikace je připravena pro definici dalšího workflow.

O aktuální volbě je uživatel informován v oblasti umístěné pod tlačítky, kde je mu oznamováno, zdali byl krok přidán, worklow spuštěno, či chybí nějaký parametr kroku.

B Povinné parametry metod

B.1 CWTPlugin

From sample
  long value

Sample count
  integer value

Channel name
  name of channel, like ”Cz”

Type of CWT
  COMPLEX_MORLET
  COMPLEX_GAUSSIAN
  GAUSSIAN
  MEXICAN_HAT
  MORLET

Lower scale limit
  double value

Upper scale limit
  double value

Step of the scale
  double value

Bandwitch constant for complex morlet cwt
  double value

Center frequency constant for complex morlet cwt
  double value

B.2 DWTPlugin

From sample
  long value

Sample count
integer value

Channel name
name of channel, like "Cz"

Type of DWT
FAST_DAUBECHIES_2
FAST_DAUBECHIES_4
FAST_DAUBECHIES_8
FAST_HAAR
FAST_SYMMLET_4

B.3 FasticalPlugin

From sample
long value

Sample count
integer value

Channel name
name of channel, like "Cz"

Number of independent components
integer value

Deflation or symmetric approach algorithm
SYMMETRIC
DEFLATION

The step size of an approach
double value

Accuracy exit condition
double value

The maximum number of iterations
integer value

B.4 FFTPlugin

From sample
long value

Sample count
integer value

Channel name
name of channel, like ”Cz”

B.5 FIRPlugin

From sample
long value

Sample count
integer value

Channel name
name of channel, like ”Cz”

Filter order
even integer value

Filter type
LP
HP
BP
BS

Window type
RECTANGULAR_WINDOW
HAMMING_WINDOW
HANN_WINDOW
HANNING_WINDOW
COSINE_WINDOW
LANCZOS_WINDOW
BARTLETT_WINDOW
TRIANGULAR_WINDOW
GAUSS_WINDOW

BARTLET_HANN_WINDOW
BLACKMANN_WINDOW
KAISER_WINDOW
NUTTALL_WINDOW
BLACKAN_HARRIS_WINDOW
BLACKMAN_NUTTALL_WINDOW
FLAT_TOP_WINDOW
BOHMAN_WINDOW
TUKEY_WINDOW
PARZEN_WINDOW
NONE

Bottom frequency value
double value

Upper frequency value
double value

Sampling frequence value
double value

B.6 MPPlugin

From sample
long value

Sample count
integer value

Channel name
name of channel, like ”Cz”

Number of iterations
integer value

Literatura

[Car()] Carmen. Dostupné z: `https://portal.carmen.org.uk/`.

[Hib()] ORM nástroj Hibernate. Dostupné z: `http://morosystems.cz/java/hibernate/ch06.php`.

[INC()] INCF Dataspace. Dostupné z: `http://www.incf.org/resources/data-space`.

[de Velde(2007)] VELDE, T. V. *Beginning Spring Framework 2*. 2007. ISBN 978-0-471-10161-2.

[ET ()] *Wicket*. ET NETERA. Dostupné z: `http://boss.etnetera.cz/cz/ke-stazeni/index.html`.

[Hall(2001)] HALL, M. *Java, servlety a stránky JSP*. 2001. ISBN 80-86330-06-0.

[Janišová(2012)] JANIšOVá, H. Workflow a elektronický oběh dokumentů. 2012. Dostupné z: `http://technet.idnes.cz`.

[Ježek(2009)] JEžEK, P. Uložiště dat a metadat EEG/ERP experimentů. 2009. Dostupné z: `http://dai.fmph.uniba.sk/events/kuz2009/prispevky-pdf`.

[Kothagal(a)] KOTHAGAL, K. *JSP and Servlets*, a. Dostupné z: `http://javabrains.koushik.org/p/jsps-and-servlets.html`.

[Kothagal(b)] KOTHAGAL, K. *Spring*, b. Dostupné z: `http://javabrains.koushik.org/p/spring-framework.html`.

[Linwood(2010)] LINWOOD, J. *Beginning Hibernate, Second edition*. 2010. ISBN 978-1-4302-2851-6.