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

Diplomová práce

Datový model pro analytické metody a systém workflow v elektrofyziologických experimentech

Plzeň, 2015 Jiří Diviš

~~zadání~~

Prohlášení

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

V Plzni dne

Jiří Diviš

Abstract

The main task of this work is to design and implement a data model for workflow system in EEG/ ERP Portal application. Design phase precedes the analytical part, that explores already existing workflow systems. Also there will be mapped the current state of the analytical methods and workflow system usability in scope of the EEG / ERP Portal application. Possibility of the application of non-relational databases will be evaluated during the model design. According to the proposal, implementation of the data model, presentation layer and a prototype system for workflow processing, that works together with EEG processor, has been created.

Abstrakt

Hlavním úkolem práce je návrh a implementace datového modelu pro systém workflow v aplikaci EEG/ERP Portál. Fázi návrhu předchází analytická část, ve které byly prozkoumány již existující systémy pro tvorbu workflow a také byl zmapován současný stav využití analytických metod a systému workflow v rámci EEG/ERP Portálu. Při návrhu modelu jsou zhodnoceny možnosti využití nerelačních databází pro daný problém. Podle vytvořeného návrhu proběhla implementace datového modelu, prezentační vrstvy a prototypu systému na zpracování workflow, který spolupracuje s aplikací EEG Procesor.

Poděkování

Rád bych poděkoval Ing. Romanu Moučkovi, Ph.D., vedoucímu své diplomové práce, za poskytnutí odborných rad, věcné připomínky a nekonečnou trpělivost při vedení této práce.

Obsah

1 Úvod

Neuroinformatika je mladý vědní obor, který se zabývá především možnostmi využití výpočetní techniky pro medicínu. Aplikací analytických výpočetních metod jsou z dat získávány nové poznatky o funkci nervového systému, mechanizmech neurologických onemocnění atd. Prolínání výzkumu v oblasti neurověd a informatiky je oboustranně přínosné, neboť jedné straně přináší nové poznatky o fungování mozku, na straně druhé pak rozvoj a vývoj výpočetních metod pro analýzy a modelování.

Vědecký tým založený Katedrou informatiky a výpočetní techniky ZČU se již několik let ve svých laboratořích zabývá výzkumem elektroencefalografie a metody evokovaných potenciálů. Při těchto experimentech je generováno značné množství výstupních dat, které je nutné uchovávat pro pozdější analýzu. V rámci výzkumu byla na katedře vyvinutá aplikace EEG/ERP Portál, která je centrálním úložištěm dat ze všech takových experimentů.

Aplikace nabízí také rozšířené možnosti pro správu experimentů, vyhledávání a repositář vědeckých článků týkajících se neuroinformatiky. Dalším logickým krokem ve vývoji je nabídnout výzkumníkům řešení, které jim poskytne možnost provést analýzu svých uložených dat pomocí speciálně navržených výpočetních procedur. Ve fázi analýzy jsou tyto procedury skládány do bloků, jejichž větvením a řetězením vznikne celý pracovní postup výpočtu. Vzniklý pracovní postup nazýváme workflow. Tato práce zavádí do EEG/ERP Portálu možnost zpracování workflow nad daty pořízenými při elektrofyziologických experimentech.

1.1 Základní principy zpracování EEG/ERP signálu

Vysvětlení základních pojmů a stručné uvedení do problematiky měření a zpracování dat v oblasti neuroinformatiky.

1.1.1 Elektroencefalografie (EEG)

EEG je diagnostická metoda používána k záznamu aktivity mozku. Tento záznam je výsledkem činnosti způsobené thalamem (hrbol mezimozkový) a neurony mozkové kůry. Takováto činnost způsobuje měřitelné změny v polarizaci elektrických potenciálů [μV] na povrchu pokožky hlavy. EEG signál vzniká měřením těchto změn v čase. Aktivita mozku je snímána sérií elektrod, pravidelně rozmístěných na hlavě měřeného subjektu. Každá elektroda generuje vlastní signál, výsledná data jsou tedy množinou signálů. V lékařské praxi je elektroencefalografie používána k diagnostice a léčení např.: epilepsie, poruch spánku či kómatu [1]. V odvětví neuroinformatiky jsou naměřená data používána pro různé pokusy, které zkoumají využitelnost EEG také v běžné praxi či průmyslových odvětvích. Na Katedře informatiky a výpočetní techniky ZČU v poslední době proběhly EEG experimenty zkoumající vliv únavy řidiče automobilu na jeho pozornost či ovládání zařízení pomocí „myšlenek“.

1.1.2 Evokované potenciály (ERP)

Evokované potenciály (z angličtiny event-related potentials) jsou významné změny v EEG signálu vyvolané nějakým vnějším podnětem. Pro nalezení ERP vln musí být subjekt opakovaně stimulován stejnými podněty. Současně se stimulem se musí do záznamu EEG vytvořit značka, která definuje čas vzniku stimulu pro pozdější vyhodnocení záznamu. Signály evokovaných potenciálů mají tvar krátkodobých vln s velmi nízkou amplitudou, jejich tvar, latence nebo čas trvání závisí na síle podnětů a duševním stavu měřeného subjektu [2]. Ve srovnání s EEG signálem, ERP vlny jsou o poznání nižší. Tyto vlny přicházejí jako pozadí pravidelné EEG aktivity - v tomto případě je EEG je pouze šumem a je třeba jej odstranit vhodným způsobem (např.: průměrováním signálu).

Techniku měření evokovaných potenciálů lze rozdělit na následující typy:

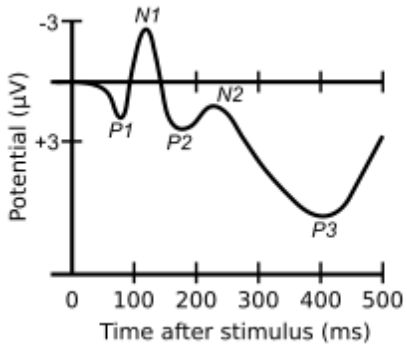
- Exogenní – reakce na fyzický stimul (charakterizován kratší dobou odezvy)
- Endogenní – vyvolaný kognitivním procesem, tedy požadavkem na rozpoznání různých stimulů (delší doba odezvy na podnět)

Další možné rozdělení lze definovat podle typu použitého stimulu:

- Zvukový – například krátké pípnutí v různých tónech či frekvencích
- Zrakový – rychlá ukázka obrázku či blikání diody různých barev.
- Somatosenzorický – reakce na různé stimuly jako jsou např.: pohyb objektu, dotyk, změna teploty apod.

1.1.2.1 Komponenty ERP

ERP vlny lze dekomponovat na jednotlivé složky [1], které jsou pojmenovány podle polarity (P – pozitivní, N – negativní, C – nestálá polarita) a podle přibližné pozice na časové ose [ms/100].



Obr. 1-Typický tvar ERP vlny [1]

Nejčastěji hledanou a nejlépe rozpoznatelnou komponentou je P3, která se vyskytuje v rozmezí 300-500 ms a dosahuje nejvyšší amplitudy ze všech komponent. Existuje řada hypotéz, které tvrdí, že právě podle pozice komponenty P3 na časové ose, lze odvodit závěry různých teorií.

1.1.3 Analýza EEG/ERP signálů

Zpracování EEG/ERP signálů je komplexní proces, kde naměření „syrových“ dat je prvním krokem daného experimentu. Dále, aby bylo možné vyčíst z těchto dat požadované výsledky, naměřená data musí projít sérií procedur na jejich úpravu a zpracování. Procedury, jejich pořadí a parametry nejsou pevně definovány, mohou se lišit pro každý specifický experiment. I přesto, že neexistuje uniformní řešení pro všechny experimenty, lze najít společné prvky a jistý „tradiční“ pracovní postup při zpracování EEG signálu.

1.1.3.1 Detekce artifaktů

Měřený signál je znehodnocen artifakty. Artifakt je charakterizován náhlým, krátkým výkyvem amplitudy. Takové artifakty vznikají např. mrknutím oka. Signál v těchto bodech a jejich blízkém okolí je odstraněn a není zahrnut v dalším zpracování.

1.1.3.2 Detekce epoch

Pro vyhodnocení naměřených dat při ERP experimentu není potřeba celý zaznamenaný signál, ale důležité jsou pouze kusy nacházející se v okolí výskytu stimulu. Epochami nazýváme právě tyto kusy. Délka epochy obvykle odpovídá délce vlny, ke které je z obou stran časové osy přidána malá rezerva.

1.1.3.3 Průměrování (epoch)

Průměrování EEG signálu obecně vede k zlepšení kvality signálu. Tento proces odstíní naměřená data od šumu a případných vlivů artefaktů. Typicky je doba ERP experimentu rozdělena do více fází a v jedné fázi se nachází více epoch. V takovém případě je možné průměrovat epochy jedné fáze a průměry fází porovnávat. Dalším případem je průměrování odpovídajících fází několika různých experimentů – tzv. „velké“ průměrování (grand average).

1.1.3.4 Baseline correction

Častým jevem při měření při dlouhých experimentech je pozvolné klesání úrovně signálu, které je zapříčiněno většinou vysycháním vodivého gelu v elektrodách či perspirací subjektu v oblasti instalace elektrod. Tzv. baseline correction úprava srovná signál na konstantní hladinu.

1.1.3.5 Peak detection

Určení vrcholu amplitudy vlny v epoše. Obvykle prováděno manuálně, u rozsáhlých experimentů může být zautomatizováno.

1.1.4 Pokročilé metody zpracování signálů

Procedury ve výše uvedeném výčtu jsou základní, obecně aplikovatelné metody pro zpracování EEG/ERP signálu. V souvislosti s analýzou ERP existují pokročilejší metody, které umožňují nejen přesnější analýzu, ale mohou být schopné autonomně detekovat a rozpoznávat jevy v reálném čase. Takové metody nacházejí uplatnění v BCI (Brain-computer interface) systémech [3]. Metody vhodné pro detekci ERP vln:

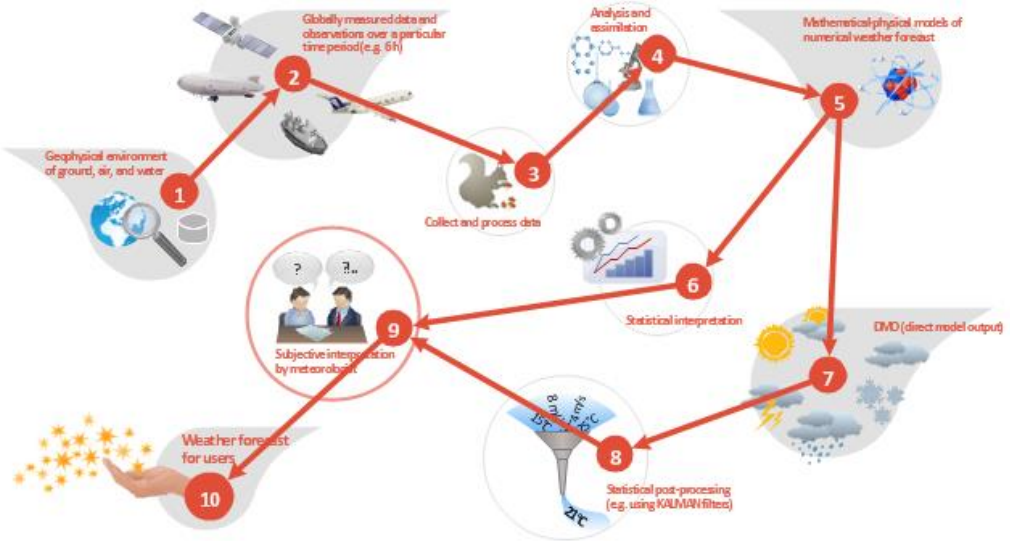
- LDA (Linear discriminant analysis)
- Fourierova transformace
- Wavelet (Vlnková) transformace
- Matching pursuit
- Hilbert–Huang transformace

1.2 Workflows

1.2.1 Pojem workflow

Workflows pocházejí z oblasti managementu podniků, kde slouží pro organizaci firemních procesů, která následně vede k lepšímu využití dostupných prostředků a ke snižování provozních nákladů firmy.

Proces symbolizuje jeden určitý problém z reálného světa. Workflow je abstrakce, která popisuje kroky potřebné pro dokončení tohoto procesu a tok informací mezi nimi. Každý krok je definován souborem činností, které je potřeba provést. Workflow určuje pořadí, ve kterém se kroky vykonávají, kroky lze vykonávat paralelně, pokud mezi nimi neexistuje závislost. Počet vykonaných kroků je konečný, workflow je obvykle zobrazené jako strom či graf, tedy jako kolekce kroků (uzlů) spojených závislostmi (hranami), které reprezentují tok dat ze vstupních portů na výstupní. Jak jednotlivé kroky, tak cesty mezi nimi mohou obsahovat vlastní atributy [4].



Obr. 2 - Příklad obecného workflow

1.2.2 Využití workflow při modelovaní business procesů

BPM (Business Process Management) je sbírka pracovních činností, které umožňují plánování a řízení podnikových procesů. Aby se daly tyto procesy efektivně využívat, je zapotřebí je jednotným způsobem popsat, implementovat, zprovoznit a sledovat. Pro tyto účely jsou vyvíjena speciální softwarová řešení.

První workflow systémy se začaly zavádět v devadesátých letech, jejich hlavním cílem bylo nahradit neefektivní plánování činností v papírové formě.

Průběhem času se objevoval software, který zaváděl implementaci pokročilých funkcí, jako jsou definice obchodních pravidel, správa zásad, visuální nástroje modelování, real-time monitorování procesů a optimalizace celého procesu. V roce 2005 Microsoft uvedl Windows Workflow Foundation, sofistikovaný programovatelný engine pro modelování procesů, který nahradil dosavadní základní systémy a do jisté míry ovlivnil trend současných systémů [5].

1.2.3 Využití workflow při zpracování EEG/ERP

Zpracování workflow se z oblasti managementu rozšířilo i do dalších odvětví a disciplín, ve kterých existuje analogie k modelování a řízení procesů. Workflow a workflow systémy má smysl používat i v oblasti encefalografie věnující se zpracování EEG/ERP signálu. Správné zpracování naměřených dat vyžaduje od výzkumného pracovníka poměrně rozsáhlou znalost této problematiky. Analytické metody mají často velké množství vstupních parametrů, které řídí a ovlivňují průběh zpracování signálu. Špatné nastavení těchto parametrů může zkreslit či znehodnotit výsledky. Pokud je v rámci pracovního postupu aplikováno více analytických metod po sobě, riziko špatného výsledku roste. K dosažení správného výsledku ale nemusí být vždy třeba jen perfektních znalostí analytických metod, ale třeba také zkušeností s měřením nebo dokonce řešení může být výsledkem náhody. Pokud se správný pracovní postup uloží ve formě workflow, je možné jej spustit opakovaně s jinými příbuznými daty nebo celý postup sdílet s méně zkušenými členy výzkumného týmu nebo dokonce měření a zpracování signálů automatizovat.

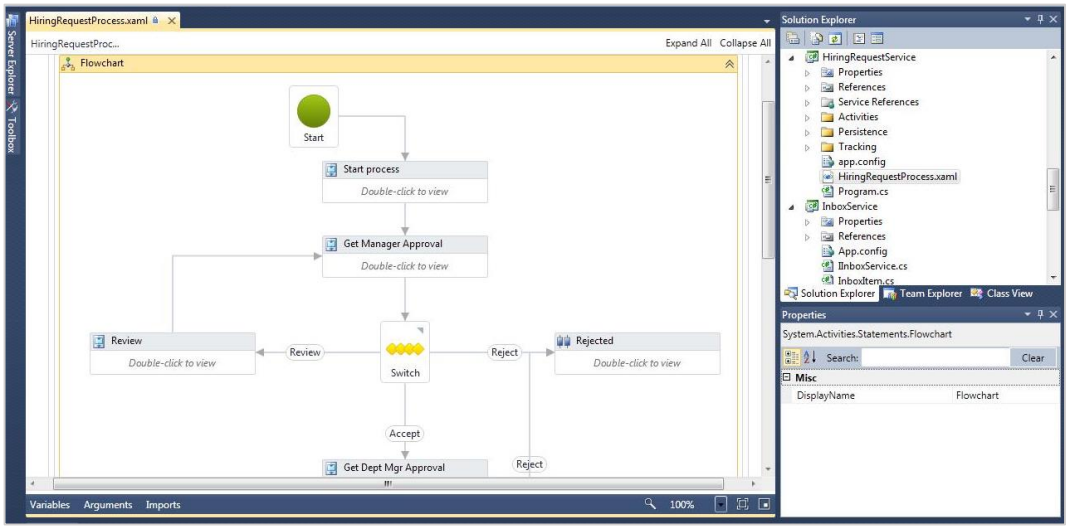
EEG signál generuje relativně rozsáhlá data, pokud je signál zpracováván přes více metod, může analýza trvat dlouhou dobu. Pokud by tato analýza byla spuštěna ve workflow systému, který zvládá zpracování akcí na pozadí (server - side), může být pro výzkumníka výhodou, že jeho stanice není blokována vlastním výpočtem.

2 Dostupná řešení pro zpracování workflow

2.1 Nástroje pro zpracování obecných workflow

2.1.1 Windows Workflow Foundation

Windows Workflow Foundation je technologie společnosti Microsoft, která poskytuje programové rozhraní (API) týkající se zpracování workflow a uživatelské rozhraní pro jeho ovládání. Workflow je zde definováno jako řada odlišných programových kroků nebo fází. Každý krok je modelován jako tzv. Activity. .NET Framework poskytuje knihovnu „aktivit“ (například WriteLine, což je činnost, která zapisuje text do konzole nebo jiné formě výstupu). V případě potřeby je možné vyvíjet vlastní „aktivity“ pro různé specifické činnosti.



Obr. 3 - Modelování workflow ve WWF [7]

Činnosti mohou být sestaveny do workflow s využitím visuálního nástroje Workflow Designer, což je „plug-in“, který standardně běží v rámci aplikace Visual Studio, ale může být integrovaný i do jiného systému. Zapouzdření programované funkce do „aktivit“ umožňuje vývojářům vytvářet lépe spravovatelné aplikace, každá vykonávaná složka totiž může být

naprogramována jako objekt Common Language Runtime, jejichž spuštění bude řízeno při běhu workflow [6].

2.1.1.1 Hlavní funkcionalita WWF

Workflow engine WWF poskytuje následující funkce:

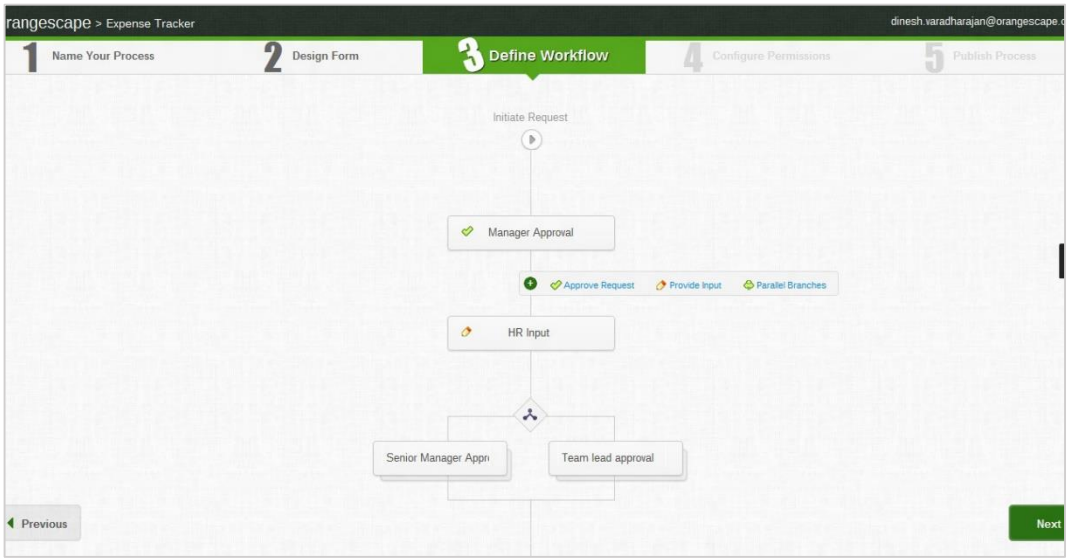
- Plánování a provádění workflow. Workflows mohou být spuštěny jedním ze tří způsobů:
  - WorkflowInvoker, který vykonává pracovní postupy na volajícím vláknu (při vytvoření workflow není vytvořené nové vlákno). To znamená, že volající proces bude čekat na dokončení workflow.
  - WorkflowApplication, který vykonává workflows v novém vlákně (tak, že volající aplikace nebude výkonově omezena při běhu workflow).
  - WorkflowServiceHost, která spustí workflow jako služby. Tyto služby obvykle používají data ze sítě jako vstupy pro obsažené „aktivity".
- Visuální editor pro modelování workflows. K dispozici jsou např. prvky Flowchart, If, Sequence, Pick, a Parallel.
- Perzistentní (přetrvávající) workflows. Při běhu si toto workflow ukládá svá data na perzistentní médium (například SQL Server) a uvolní tak neaktivní workflow z paměti. Workflow může být opět načteno po uplynutí stanovené doby, nebo pokud je přijata zpráva. Odstraněním nečinných workflow z paměti, může daný workflow engine výrazně zvýšit počet aktivně zpracovávajících workflows, čímž se zvyšuje škálovatelnost.
- Správa dat nutných pro běh „aktivit". Systém řídí data, která zpracovávané činnosti při svém výpočtu potřebují.
- Umožňuje rozšiřitelnost ve formě Workflow Extensions.

2.1.1.2 Tvorba workflow ve WWF

Workflows jsou ve visuálním editoru aplikace vytvářeny v jazyku XAML (Extensible Application Markup Language) nebo přímo naprogramovány v jazyce .NET. V případě použití editoru, se „aktivity“ vytvářejí na plátně metodou „drag and drop“. Pokud je workflow programováno, „aktivity“ jsou instancemi CLR objektů a sestaveny do kolekcí (Sequence, FlowChart,..).

2.1.2 KiSSFLOW

KiSSFLOW (Keep it Simple & Smart Workflow) je software pro automatizaci workflow založený na cloudové technologii. KiSSFLOW nabízí webové rozhraní a mobilní aplikace, které poslouží k organizaci strukturovaných workflow procesů dané organizace. Systém slouží k organizaci procesů, které jsou primárně vykonávané lidmi (žádost na IT service, žádost o podpis či schválení dokumentu, povolení k odjezdu apod.).



Obr. 4 - Modelování workflow v aplikaci KiSSFLOW[8]

Systém pro své fungování využívá integraci do Google Apps (např. Docs, Contacts, Mail). Vytváření workflow není podmíněno žádnou technologickou znalostí uživatele, editor worfklow obsahuje průvodce, který při vytváření

workflow uživatele v pěti krocích naviguje. Aplikace rozlišuje role uživatelů, pro které jsou daná workflow viditelná. [8]

2.2 Nástroje pro zpracování vědeckých workflow

Systémy pro zpracování vědeckých workflow jsou speciálním případem workflow enginů, které nahrazují obvyklé business procesy analytickými výpočty.

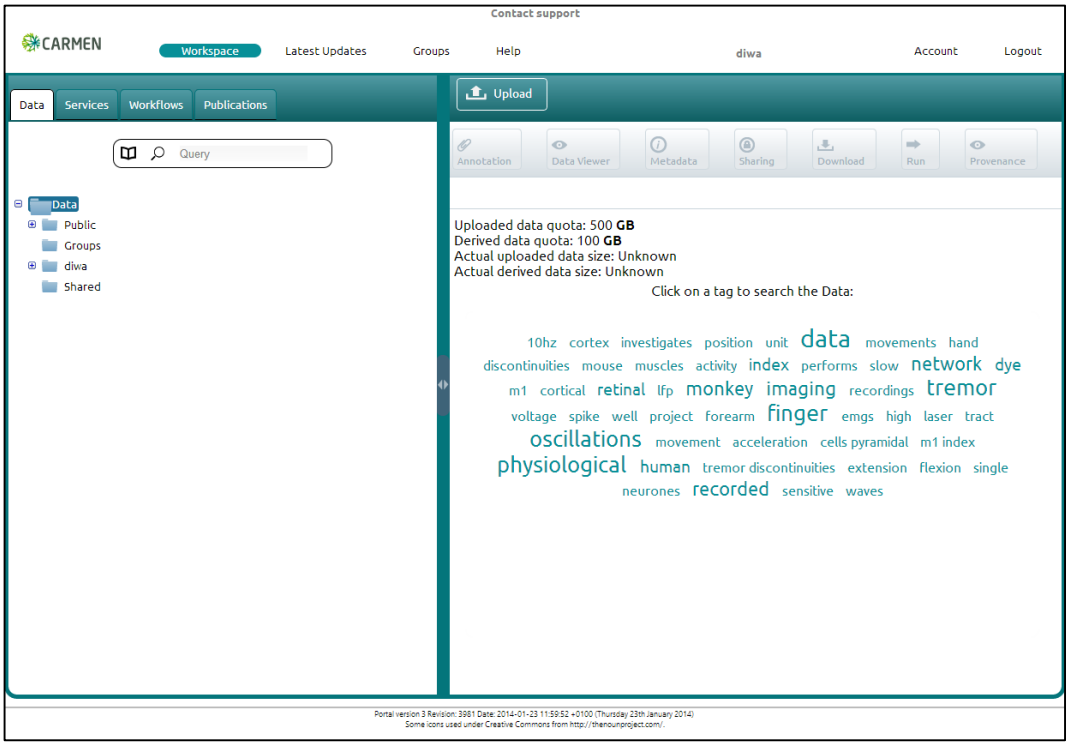
2.2.1 Taverna, E-science, openVIBE

Vlastnosti a funkčnost systémů pro práci s workflow Taverna, E-science a openVIBE byly zkoumány a detailně popsány v práci Jana Štěbetáka [9]. Citované přeložené zhodnocení:

Výše zmíněné systémy jsou navrženy čistě pro vědecké účely. Poskytují možnosti tvorby workflow napříč mnoha vědeckými odvětvími včetně neuroinformatiky a zpracovávání EEG/ERP experimentů. Všechny popsané systémy zahrnují typovou kontrolu parametrů, která zajišťuje správnou návaznost po sobě jdoucích bloků. Nicméně, procedury používané při analýze EEG/ERP jsou specifické svými požadavky jak na správnou syntaxi, tak i na správnou sémantiku zvolených vstupů či výstupů jednotlivých bloků. Například pouze podmnožina výsledků předchozí procedury může navazovat na vstup v pořadí následující procedury. Podobné případy tyto systémy neřeší. Pro dobře fungující EEG/ERP workflow, není zajištění pouze syntaktické návaznosti dostačující. Použité procedury musejí být správně propojené i z hlediska sémantiky. Z uvedených systémů se pouze Taverna zabývá řešením sémantiky. Jedná se o formální sémantiku, ve které jsou definovaná omezení pro vstupní a výstupní parametry (povolené hodnoty). Nicméně sémantika navazujících metod (zda spojení dává či nedává smysl) zde není vyřešená. Všechny tyto aplikace jsou navrženy pro přidávání nových procedur (zahrnující i oblast EEG/ERP). Tyto procedury jsou obvykle uloženy v centrálním úložišti systému. Uložiště E-Science je založeno na cloudové technologii. Má vlastní úložiště, ale zároveň je otevřené pro přidávání procedur třetích stran.

2.2.2 Carmen Portal

Systém CARMEN je navržen tak, aby lidem pohybujícím se v oblasti neuroinfromatiky umožnil sdílet data programy (služby) v rámci neurofyziologických experimentů. Portál Carmen poskytuje pro přístup do sytému webové uživatelské rozhraní dostupné v internetovém prohlížeči. Umožňuje uživatelům přístup k výpočetnímu jádru a zdrojům v úložišti [10].



Obr. 5 - Náhled na aplikaci Carmen

2.2.2.1 *Hlavní funkcionalita*

Tento portál umožňuje registrovaném uživateli následující možnosti:

- Uložení dat naměřených při experimentech či simulacích
- Uložení a spuštění analytického kódu
- Uložení metadat spojených s výpočtem
- Vyhledávání napříč kolekcemi dat
- Kontrolované a zabezpečené možnosti sdílení dat, analytického kódu či výsledků
- Využití standardních formátů pro výměnu dat

2.2.2.2 *Workflow v aplikaci Carmen*

Nahrávání dat

Podle tvrzení vývojářů aplikace umožňuje na interní server nahrát až 500 GB vstupních dat, pro výsledky je přidělená kapacita 100 GB. Reálný limit těchto optimistických hodnot nebyl vyzkoušen. Spolu s vstupním daty lze nahrát dodatečné informace (metadata) vázané k experimentu. Data je možné dělit do několika skupin:

- Public – veřejná skupina, data jsou přístupná všem registrovaným uživatelům
- Group – data jsou zveřejněna v rámci uživatelské skupiny
- Private – pouze vlastník vidí tato data

Analytické procedury jak služby

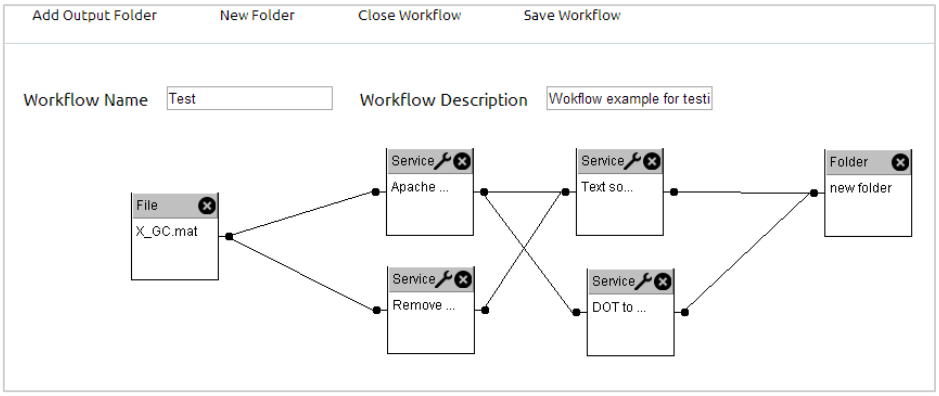
Systém umožňuje uživatelům ukládat a archivovat analytické procedury v podobě webových služeb. Carmen systém by měl umět převést kód psaný v laboratorním prostředí (v jazycích, jako je Matlab, R, Python, JAVA), do služeb, které jsou k dispozici ostatním uživatelům. Stejně jako u dat a metadat, i zde existuje podobná bezpečnostní logika ovlivňující, kdo může vidět nebo spustit služby. Služby a algoritmy nelze stáhnout, zůstanou pod kontrolou původního autora. V současné době, služby nemohou být přímo nahrané od všech koncových uživatelů, ale zřejmě jen od vybraných uživatelů – beta testerů či vývojářů.

Nicméně na portálu existuje řada služeb, které jsou veřejně k dispozici. Mezi nimi existuje i několik běžných procedur pro zpracování neurofyziologických experimentů (např. datové filtry, detekci neuronových špiček a jejich řazení). Bohužel v době testování nebyl dostupný celý seznam veřejných procedur, procedury bylo možno pouze vyhledávat podle klíčových slov. Například po zadání klíčového slova “text“ bylo zobrazeno k dispozici 17

různých procedur týkajících se zřejmě zpracování textů, při zadání hesla “eeg“ byl zobrazen pouze jeden výsledek.

2.2.2.3 *Modelování workflow*

Od března 2014 je v rámci portálu Carmen implementováno i rozhraní pro modelování vlastních workflow. Je možné neomezeně vázat vstupy a výstupy analytických procedur, aplikace nekontroluje žádnou syntaktickou či dokonce sémantickou návaznost po sobě jdoucích procedur.



Obr. 6 - Příklad vytvořeného workflow v aplikaci Carmen

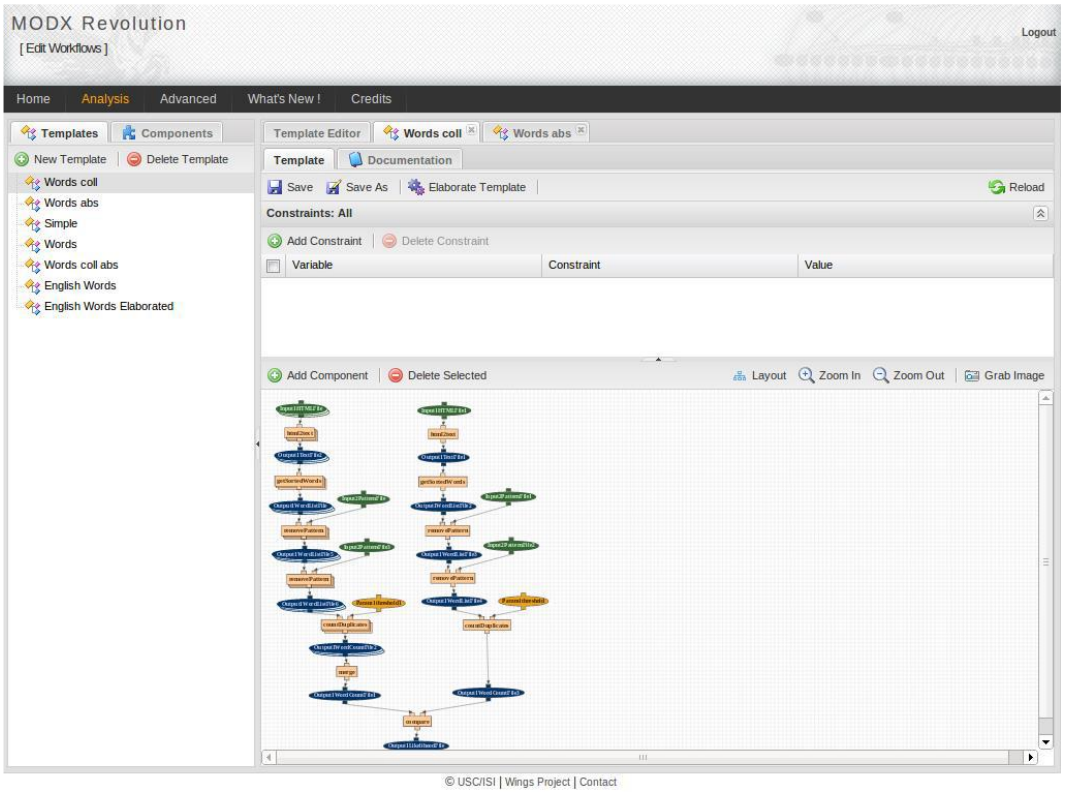
2.2.2.4 *Spuštění workflow*

Vytvořené workflow lze spustit na pozadí, uživatel může tedy během výpočtu zavřít prohlížeč a výsledky si prohlédnout později. Pokud uživatel správně uhodne návaznost použitých procedur, spuštěné workflow úspěšně proběhne. Uživatel je o průběhu informován v sekci „Log“. Výstupní data jsou uložená v záložce data, zařazená do složky „Output“. Pokud během výpočtu nastane chyba, ve složce „Log“ je workflow označené jako chybně provedené, bez dalších informací o chybě.

2.2.3 Wings

Wings je sémantický workflow systém, určený pro návrh vědeckých výpočetních experimentů. Unikátním rysem systému je to, že jeho do workflow lze začlenit sémantická omezení týkající se souborů a workflow komponent. Správnost vytvořeného pracovního postupu lze tedy ověřit. WINGS je postaven

na otevřených webových standardech z World Wide Web Consortium (W3C), jako jsou webové Ontology Language (OWL), Resource Description Framework (RDF) a dotazovací jazyk SPARQL. Jádro ontologie je definováno pomocí OWL, zatímco šablony workflow, žádosti a kandidáti jsou popsány v RDF [11].



Obr. 7 - Náhled na aplikaci Wings

2.2.3.1 *Hlavní funkcionalita*

- Workflow je systém, který byl postaven na základech otevřených webových standardů
- Aplikace je instalována do prostředí uživatele. Nejedná se o žádnou cloudovou službu.
- Umožňuje uživateli použít vysokou abstrakci pro popis dat, procedur a celých workflow.
- Do workflow je možné zavést sémantická omezení.

2.2.3.2 *Workflow v aplikaci Wings*

2.2.3.2.1 *Nahrávání dat*

Wings také nabízí možnost nahrání dat do aplikace, takže může uživateli sloužit jako repositář (není potřeba data uchovávat na ftp serveru). Bohužel na rozdíl od systému Carmen není aplikace orientovaná na sdílení zdrojů mezi uživateli. Vstupní data mohou být rozdělená do skupin, ale tyto skupiny jsou stejné pro všechny uživatele.

2.2.3.2.2 *Komponenty workflow*

Ve standardním balíčku dodávaném s instalací jsou pouze ukázkové komponenty (z oblasti zpracování textu). Uživatel si musí všechny výpočetní procedury vytvořit sám. Kostra každé procedury je definovaná jako unix shell skript. V této kostře je definováno navázání vstupů a výstupů a bod spuštění procedury.

2.2.3.2.3 *Modelování workflow*

Wings systém je také vybaven interaktivním editorem workflow. Editor napomáhá uživateli při modelování workflow tím, že v reálném čase kontroluje návaznost jednotlivých komponent. Například nelze spojit vstup s nekompatibilním výstupem. Pokud je návrh hotový, systém před spuštěním vytvořeného postupu zkontroluje sémantické návaznosti.

2.2.3.2.4 *Spuštění workflow*

Po spuštění workflow může uživatel v aplikaci pozorovat průběh zpracování. Po dokončení výpočtu je zobrazen návratový stav a log. Výstupní data a mezivýsledky jsou přehledně uspořádaná do struktury stromu.

2.3 Zhodnocení zkoumaných systémů

Windows Workflow Foundation je rozsáhlým workflow managerem pokrývajícím značný rozsah problematiky modelování workflow, která jsou

složená s business procesů. Aplikace je ale schopná se adaptovat i na procesy mimo oblast managementu. Nabízí propracovaný výpočetní engine, který je stavěný na workflow opravdu složitých procesů, obsahuje obslužné rutiny, které se provedou v případě selhání procesů. Modelování workflow není jednoduché, ani nijak uživatelsky pohodlné.

KiSSFLOW je také příkladem systému pro modelování procesů ve firmách, ale je přesným opakem WFF. Vyznačuje se uživatelskou přívětivostí, přehledností, mobilitou a celkovou jednoduchostí.

Carmen je začínající projekt, který je neustále ve fázi vývoje, v době testování nebyla aplikace vhodná na žádné reálné použití, kromě úložiště dat.

Wings je zajímavý program, bohužel k jeho poměrně složité instalaci nejsou dostupné potřebné návody. Wings má na svých internetových stránkách připravený sandbox, ve kterém je možnost vyzkoušet nasazenou aplikaci. V tomto sandboxu vše funguje dobře, i přesto (možná proto) že je zde starší verze softwaru. Bohužel nainstalovanou aplikaci na vlastním počítači se nepovedlo uvést do bezchybného stavu, důležité prvky aplikace nefungovaly. Po analýze tohoto programu byla odhalena další úskalí. Celý projekt je slepen z různých technologií (PHP Redakční systém + JAVA + shell skripty), jeho případná integrace do EEG/ERP portálu by byla složitá a funkčnost nezaručená. Naopak velice precizně působila část týkající se sémantiky workflow.

2.4 Závěr analýzy dostupných workflow systémů

Cílem analýzy již existujících systémů bylo pokusit se najít vhodný dostupný prostředek pro vytváření workflow, který by umožnil zpracování elektrofyziologických dat uložených v aplikaci EEG/ERP Portál. Podle zaznamenaných výsledků se vhodné existující řešení nepodařilo najít. V úvahu připadal kompromis při použití systému Wings, který vyhověl v největším počtu hledaných parametrů. Integrace systému Wings se ale nepovedlo docílit

z důvodů časové náročnosti procesu, který navíc nezaručoval úspěšný výsledek, protože výchozí aplikace nefungovala podle předpokladů. Vytvořil se tak prostor pro implementaci vlastního řešení, které bude od začátku stavěné podle specifikovaných požadavků.

Systém	Použitelný pro vědecké účely	Implementace vlastních procesů	Založený na Java technologiích	Integrovatelný do vlastní aplikace	Webová aplikace	Vhodné uživatelské rozhraní	Možnost implementovat sémantická omezení	Otevřený software
WFF	Částečně	Ano	Ne	Částečně	Ne	Ne	Ne	Ne
KiSSFLOW	Ne	Ano	Ano	Ne	Ano	Ano	Ne	Ne
Carmen	Ano	Částečně	Ano	Ne	Ano	Ano	Ne	Ne
Wings	Ano	Ano	Částečně	Ano	Ano	Ne	Ano	Ano
Taverna	Ano	Ano	Neznámo	Ne	Ne	Ano	Ano	Ano
E-Science	Ano	Ano	Částečně	Ne	Ano	Ano	Ne	Ne
OpenViBE	Ano	Ano	Ano	Ne	Ne	Ano	Ne	Ano

Tab. 1 - Zhodnocení testovaných systémů

3 EEG/ERP Portál

EEG/ERP Portál je interní název pro EEGbase - aplikaci výzkumné skupiny založené na Katedře informatiky a výpočetní techniky ZČU. Skupina se zabývá experimenty v oboru neuroinformatiky. Pomocí definovaných scénářů jsou osobám vystaveným specifickým vlivům měřeny evokované potenciály. Tato měření generují značné množství dat, které je nutné pro účely výzkumu správně ukládat k pozdější analýze a zpracování. Samotné měření provádí větší množství osob, které specifikace scénářů sdružují do skupin, a proto je na portále implementován hromadný přístup k systému pro ukládání a správu experimentu. Aplikace je využívána hlavně pro experimenty v rámci ZČU, ale počítá s jejím globálním rozšířením. V EEG/ERP Portálu se nyní zavádí, funkce umožňující licencování experimentů. EEG/ERP Portál je aplikací, která si klade za

cíl být centralizovaným místem pro odbornou komunitu lidí, která se aktivně zajímá o obor encefalografie.

3.1 Hlavní funkce EEG/ERP Portálu

V současné době je možné v aplikaci provádět následující akce:

- Vytváření a správa uživatelských účtů
- Vytváření výzkumných skupin a jejich organizaci
- Ukládání, analýza a zpracování dat a metadat experimentů
- Sdílení dat a metadat mezi uživateli a výzkumnými skupinami
- Přidávání článků a jejich komentovaní pro uživatele a skupiny
- Vyhledávání v EEG databázi
- Historie pro administrátora skupiny nebo systému
- Přihlášení do Portálu pomocí sociálních sítí Facebook a LinkedIn

3.2 Architektura

Aplikace EEG/ERP Portál je založena na platformě Java EE, běžící na webovém serveru a servletovém kontejneru Jetty 8.0. Aplikace využívá návrhu třívrstvé architektury, je rozdělená na datovou, aplikační a prezentační vrstvu. Jádro aplikace využívá framework Spring, který je použit pro správné propojování jednotlivých logických komponent aplikace. Prezentační vrstva je implementována s využitím frameworku Apache Wicket v aktuální verzi 6. Ukládání dat je zajištěno objektově-relačním mapováním, které poskytuje framework Hibernate.

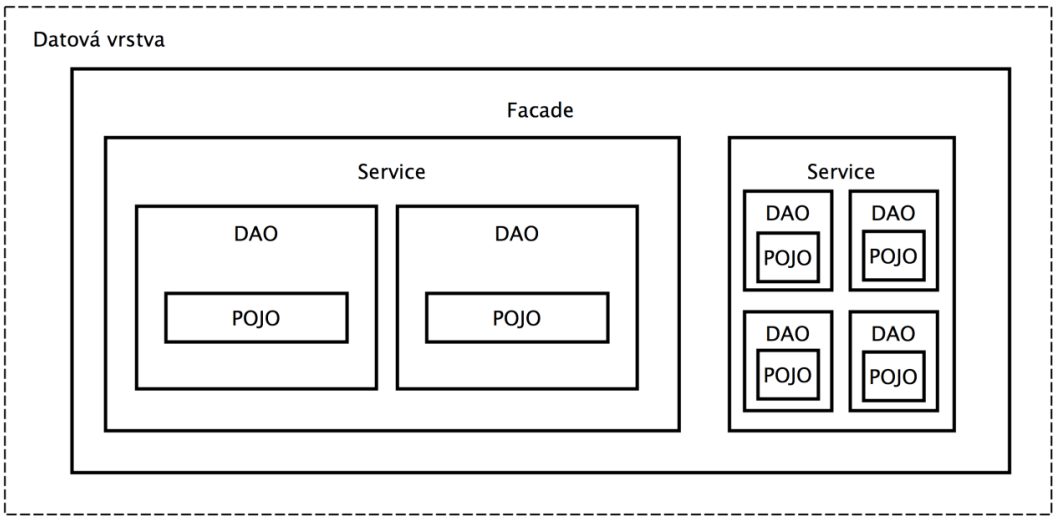
3.3 Datová vrstva

Původně EEG/ERP portál ukládal data do databáze Oracle 11g, která však přestala svými licenčními podmínkami vyhovovat. Datová vrstva tedy prošla důslednou restrukturalizací, kterou v rámci své diplomové práce provedl Martin

Bydžovský [12]. Nevyhovující relační databáze Oracle byla nahrazena za PostgreSQL, dále byla přidána nerelační databáze Elasticsearch. Celé databázové schéma bylo lehce upraveno, data nevhodná pro uspořádání v relačním modelu byla převedena do Elasticsearch.

3.3.1 Struktura datové vrstvy

Celá datová vrstva jde rozdělit na následující dílčí součásti.



Obr. 8 - Model datové vrstvy EEG/ERP Portálu [12]

3.3.1.1 POJO Entity

Třídy POJO (Plain Old Java Object) jsou obecně Java třídy, které splňují určité specifické podmínky. Jedná se např. o bezparametrický konstruktor nebo žádné výkonné metody (s výjimkou getterů a setterů). V aplikaci se používají pro reprezentaci persistentních entit, které se ukládají do databáze. Hibernate se stará o mapování jejich atributů na konkrétní sloupce v tabulkách a jejich automatické načítání a zápis z/do databáze. Samy o sobě nemají žádnou funkcionalitu, pouze reprezentují data.

3.3.1.2 DAO

DAO, neboli Database Access Object, je skupina tříd, které jsou svázané s (většinou jedním) POJO objektem. Obsahují jednak metody pro zápis, aktualizaci a mazání záznamů v databázi (vztahující se právě k tomuto POJO),

tak metody pro jejich získávání, např. na základě id, filtrování podle hodnot atributů a podobně. DAO vrstva se stará o vytváření POJO objektů.

3.3.1.3 Service

DAO třídy obsahují metody pro práci s jednou entitou a provádějí z hlediska fungování celé aplikace velmi jednoduché a dílčí operace. A právě z tohoto důvodu existuje servisní vrstva, která má k dispozici reference na několik DAO objektů a je schopná provádět komplexnější operace. Např. registrace uživatele znamená z pohledu celé aplikace tyto kroky [12]:

- zašifrovat jeho heslo
- uložit objekt uživatele do databáze
- přidělit mu roli běžného uživatele
- pokud byla součástí registrace pozvánka do nějaké výzkumné skupiny, připojit ho do ní
- odeslat potvrzovací email

V neposlední řadě tato vrstva obstarává transakce, které jsou drženy na úrovni service objektu. Celá operace prováděná nad service objektem se potom jeví navenek jako atomická.

3.3.1.4 Facade

Facade plní podobnou úlohu jako vrstva service. Jde o další úroveň abstrakce, která může provádět velmi rozsáhlé operace s daty. Přitom ke své funkčnosti může využívat jednu nebo více service objektů.

3.3.2 SQL vs. NoSQL

Základním prvkem tradičních relačních (SQL) databází jsou relace – databázové tabulky. Jejich sloupce se nazývají atributy, řádky tabulky jsou pak záznamy. Atributy mají určen svůj konkrétní datový typ a doménu, což je množina přípustných hodnot daného atributu. Řádek je řezem přes sloupce

tabulky a slouží k vlastnímu uložení dat. Pojem „relační databáze“ souvisí s teorií množin. Každá konkrétní tabulka totiž realizuje podmnožinu kartézského součinu množin přípustných hodnot všech sloupců – relaci.

NoSQL (vykládáno jako „Not only SQL“) je databázový koncept, ve kterém se pro datové úložiště i pro operace nad daty používají jiné prostředky než tabulková schémata tradičních relačních databází. Hlavní motivací k tomuto přístupu jsou především jednoduchost designu, horizontální i vertikální škálovatelnost a jemnější kontrola dostupnosti. Díky odlišné struktuře ukládání dat (např. stromová, grafová) oproti RDBMS, je i algoritmická složitost pro různé operace odlišná. Obecně se vhodnost aplikace daného typu databáze liší podle řešeného problému [13].

3.3.2.1 Typy NoSQL databází

Základní klasifikaci NoSQL databází lze provést podle typu ukládaných dat [14]:

- Dokumentové - základním prvkem dokumentových databází je pojem dokumentu. Ten obsahuje částečně strukturovaná data, která čítají libovolné položky (klíče). Nad těmito klíči se pak dají vytvářet indexy a lze podle nich dokumenty vyhledávat. Pro kódování dokumentů se používá spousta formátů, např. JSON, XML, YAML. Do této kategorie patří systémy MongoDB, Elasticsearch, Neo4J, CouchDB, Solr a další.
- Klíč-hodnota - velmi jednoúčelové databáze, které jsou pro daný úkol vysoce optimalizované. Data lze získávat jen pomocí primárního klíče. Často se snaží veškerá data držet v RAM, čímž dosahují velmi vysokých rychlostí zápisu a čtení. Zástupci této kategorie jsou systémy Memcached, Redis.
- Grafové - databáze používají grafové struktury jako uzly a hrany k reprezentaci dat. Uzly jsou přímo propojeny se svými sousedními prvky, takže není třeba dělat žádné JOIN operace a prohledávat index.

Využívají se například k reprezentaci dat na sociálních sítích, v mapách nebo třeba síťových topologiích.

3.3.2.2 Srovnání SQL a NoSQL databází

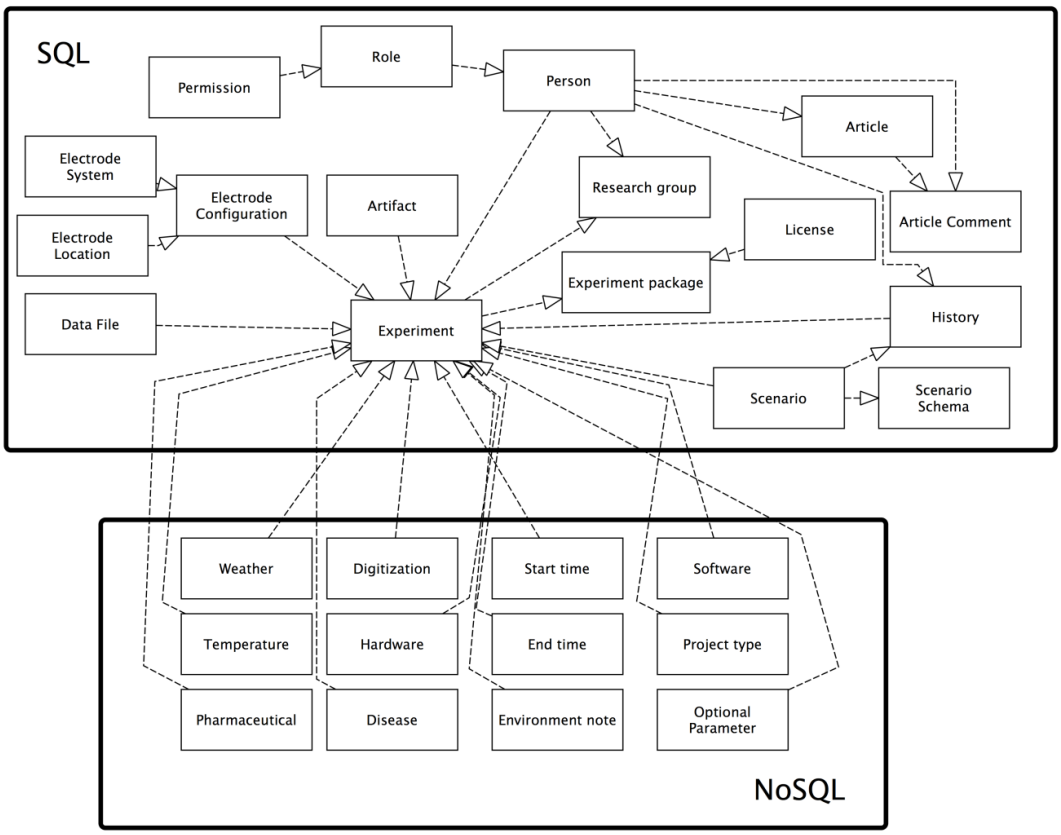
Vlastnost	NoSQL	SQL
Styl ukládání dat	Dokumenty, dvojice klíč-hodnota, grafové struktury…	Tabulky.
Organizace dat	Dynamické schéma nestrukturovaných dat.	Předem dané, jasně definované schéma.
Škálování (zvyšování výkonu)	Horizontální – vyššího výkonu se dosahuje přidáním více běžících serverů.	Vertikální – zvyšuje se výkon běžícího serveru (více RAM, silnější procesor, SSD disky…).
Zpracovávaná data	Vhodné pro hierarchická data s různě zanořenými položkami na způsob JSON nebo XML dokumentů	Komplexní data s množstvím složitých vazeb mezi jednotlivými tabulkami. Méně vhodné pro hierarchicky vrstvená data.
Dotazovací jazyk	Každá NoSQL databáze používá vlastní dotazovací jazyk. Většinou pro získávání jasně daných dat z databáze podle jednodušších kritériích.	Používá se standardizovaný SQL pro pokládání složitých dotazů. Vhodný k tvorbě různých komplexních projekcí nad zdrojovými daty.
Provázanost dat	Protože neexistuje efektivní možnost spojování souvisejících entit (dokumentů), související data se často kopírují jako vnořené sub-dokumenty v rámci ukládané entity.	Relace se definují cizími klíči – záznam obsahuje referenci na související entitu v jiné tabulce.
Bezpečnost dat	V (distribuovaných) NoSQL databázích nelze zajistit požadavky ACID známé z relačních databází.	Podporují tzv. ACID přístup – Atomicita operací (uzavřených v transakci), Konzistence – v každý okamžik jsou data v bezchybném stavu, Izolovanost – jak (ne)jsou transakce viděny okolnímu prostředí a Trvanlivost.

Tab. 2 - Porovnání vlastností relačních a nerelačních databází [12]

3.4 Rozdělení dat mezi relační a nerelační části databáze.

Podle výše popsaných vlastností a rozdílů mezi typy databází jsou data EEF/ERP Portálu rozdělená mezi relační a nerelační část datového úložiště.

Důležitá kmenová data jsou ukládána do relační databáze. Jedná se o seznam uživatelů, uživatelských skupin, jejich přístupových práv, články včetně komentářů, měřené experimenty, datové soubory, licence. Do nerelační části byly přesunuty různé doplňující parametry a metadata vztahující se k experimentům (např. popis schématu zapojení elektrod). Obecně se jedná o data a datové struktury, u kterých se klade důraz na flexibilitu, jedná se tedy o dynamické entity, jejichž udržitelnost by byla v budoucnu nákladná.

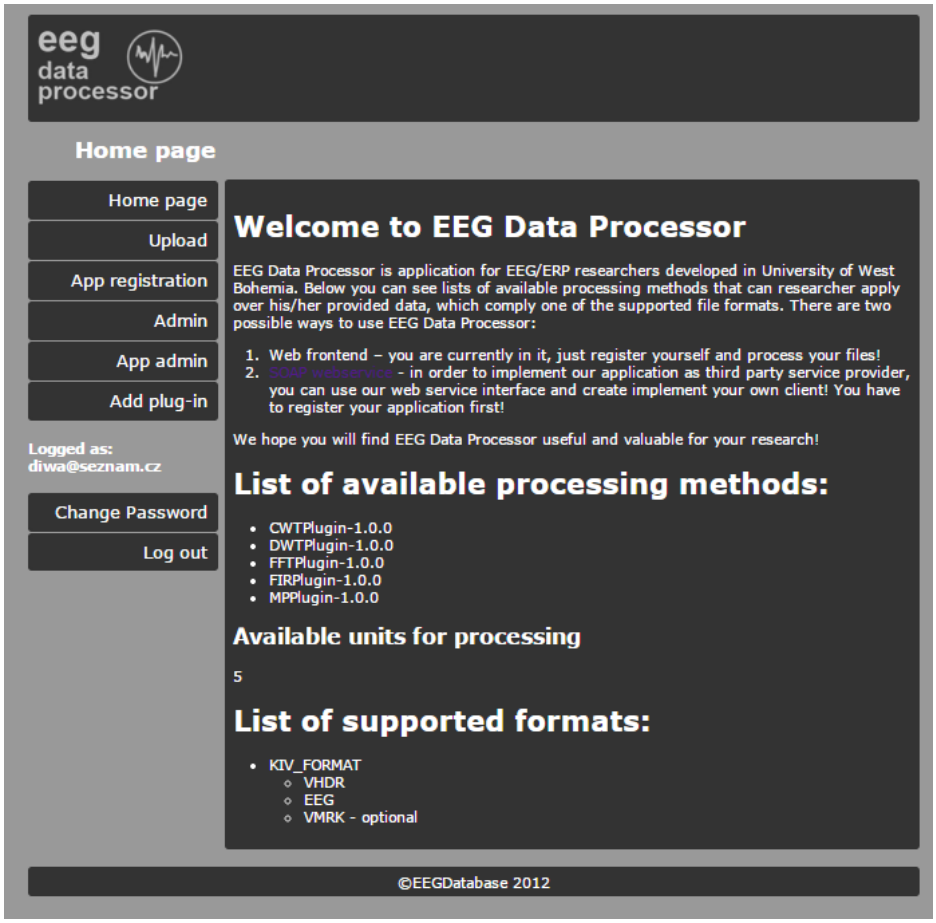


Obr. 9 - Zjednodušené schéma relační databáze [12]

4 EEG data processor

Pro analýzu naměřených dat existuje i vlastní řešení vyvinuté členy výzkumné skupiny KIV ZČU. EEG data processor je webová aplikace psaná v jazyce Java, využívá Spring a Hibernate frameworky. Aplikace nabízí registrovaným uživatelům několik analytických procedur pro zpracování jejich dat. Analytické procedury jsou do aplikace zaváděny jako tzv. pluginy, což

umožňuje za běhu vkládat do aplikace nové procedury. Pro zpracování dat je připravená SOAP webová služba, pomocí které je možné spouštět procedury i vzdáleně.



Obr. 10 - Náhled na aplikaci EEG data processor

4.1 Dostupné procedury

V současné době je implementováno šest různých procedur:

- CWTPlugin-1.0.0
- DWTPlugin-1.0.0
- FFTPlugin-1.0.0
- FIRPlugin-1.0.0
- MPPlugin-1.0.0

4.2 Princip přidávání pluginů

Klíčovou činností webové aplikace EEG data processor je provádění výpočtů nad vybranými daty zvolenými výpočetními metodami. Aby bylo možné aplikaci dále rozšiřovat a umožnit uživatelům přidávat další výpočetní metody, je nutné definovat rozhraní, které deklaruje jisté požadavky na formu pluginu.

Každá nově vznikající metoda může provádět libovolné výpočty a operace. Aby bylo možné zakomponovat ji do výpočetního portálu, a umožnit její spouštění musí být splněna následující kritéria:

1) výpočetní metoda musí mít v (uživatelem) zvolené třídě právě jednu vstupní metodu
2) vstupní metoda musí být návratového typu *void*
3) prvním parametrem vstupní metody musí být pole typu *double* – neboť data získávaná ze vstupních souborů měření obsahují právě pole tohoto typu, prostřednictvím tohoto pole budou předána všechna data určená ke zpracování
4) druhým parametrem vstupní metody musí být *FileOutputStream* (z balíku *java.io*), který ponese otevřený stream určený pro zápis výstupních dat metody
5) dále může následovat libovolný počet parametrů pro konfiguraci výpočetní metody
6) vytvořenou výpočetní metodu je potřeba se všemi třídami sestavit jako jeden JAR soubor
7) aby bylo možné metodu následně používat, je nutné přidat k pluginu soubor settings.properties, který obsahuje popis výpočetní metody

4.2.1 Omezení vstupních parametrů výpočetních metod

Parametry výpočetních metod musí splňovat následující podmínky:

1) počet parametrů vstupní metody výpočetní metody není shora omezen
2) parametry mohou být pouze následujících datových typů: *boolean*, *int*, *long*, *double*, *float*, *String* a *enum* (výčtový typ)
3) aby bylo možné předat uživateli informaci o významu parametrů metody, je nutné každému parametru přidat anotaci *@Param* z balíku *cz.zcu.kiv.eeg.methods.invoker* (toto anotační rozhraní obsahuje 3 atributy typu *String*, jež ponesou popisnou informaci)
    - atribut *description* slouží ke stručnému popisu významu parametru
    - atribut *possibleValues* slouží k zadání informace o omezení přípustných hodnot pro tento parametr (v případě dat.typu *enum*, je informace ignorována a uživateli se zobrazí celý výčet hodnot daného *enum* parametru)
    - atribut *name* není v současném stavu využit

4.2.2 Požadavky konfiguračního souboru

Požadavky se rozumí zadání nutných údajů do souboru *settings.properties* umístěného v adresáři s pluginem.

- první řádka obsahuje údaj o názvu algoritmu: *algorithmName=<název>*
- druhá řádka obsahuje údaj o autorovi: *author=<jméno>*
- třetí řádka obsahuje cestu ke vstupní třídě výpočetní metody: *main.class=<balík.třída>*
- poslední řádek obsahuje název vstupní metody ve výše uvedené třídě: *main.method=<název_metody>*

4.3 Vzdálené volání procedur přes SOAP web službu

Po zaregistrování uživatele (jeho aplikace) je možné používat pro přístup k procedurám EEG data processoru připravenou webovou službu s následující mi metodami:

- availableProcessingUnits – poskytuje informace o počtu dostupných jednotek pro výpočet (EEG data processor disponuje omezeným počtem současně spuštěných procesů)
- getAvailableMethods – vrací seznam aktuálně dostupných procedur
- getMethodParameters – dotaz na vstupní parametry konkrétní metody
- processData – volání zvolené procedury, vstupní data jsou předána jako parameter

5 Návrh architektury nástroje pro zpracování workflow v EEG/ERP Portálu

EEG/ERP Portál nyní plní převážně funkci datového úložiště experimentů, tedy hlavně naměřených EEG dat. Tato data jsou z velké většiny uchovávána jako „syrová“, dá se tedy předpokládat, že v budoucnu dojde k jejich zpracovávání. Pokud by se uživatel rozhodl svá data zpracovávat, bude muset na server uložené soubory stáhnout a poskytnout je jiné aplikaci, ve které data zpracuje. Pokročilejší možností pro zpracování uživatelských experimentů je využití implementované webové služby, která může poskytovat tato data aplikacím třetích stran. Z uživatelského hlediska zpracování jeho dat není úplně pohodné (dlouhé stahování dat, nutná registrace do další aplikace), z pohledu Portálu je tento přístup také nevýhodný, protože jednak zatěžuje server vysokým provozem a hlavně, pokud by cizí aplikace pro zpracování dat nabízela podobnou službu jako Portál, jsou tato cenná data „rozdávána“ ve prospěch cizí aplikace, v hraničním případe může nastat i odliv uživatelů. Jelikož výzkumný

tým KIV ZČU může využívat i aplikaci EEG data processor, nabízí se vlastní řešení nástroje pro zpracování workflow elektrofyziologických experimentů, které vznikne spoluprací obou těchto aplikací.

5.1 Požadavky na funkčnost workflow nástroje

5.1.1 Využití již implementovaných procedur

Workflow vytvořená v aplikaci EEG/ERP Portál budou pro výpočty používat procedury implementované výzkumným týmem KIV ZČU.

5.1.2 Možnost přidání uživatelských procedur

Seznam dostupných procedur musí být rozšiřitelný. Nové procedury půjde registrovat za běhu aplikace. Procedury budou aplikaci EEG/ERP Portál poskytovány jako služby, samotná aplikace bude průběh výpočtu kontrolovat, ale nebude jednotlivé výpočty provádět. Pro výpočty je navržena aplikace EEG data processor.

5.1.3 Uložení workflow, možnost editace již uložených

Registrovaní uživatelé budou mít možnost své modely workflow ukládat do databáze aplikace, bude zajištěn přístup k jejich pozdějším úpravám.

5.1.4 Spuštění workflow

Modely workflow vytvořené v aplikaci bude možné spustit a pozorovat výsledky. Proces zpracování daného workflow nesmí blokovat činnost uživatele při práci s aplikací, tzn. výpočet bude prováděn na pozadí aplikace. Uživatel bude moci pozorovat průběh výpočtu a po dokončení bude o výsledku informován.

5.1.5 Sdílení uživatelských workflow

Uživatel bude moci sdílet sebou vytvořené modely workflow. Budou zavedeny dvě úrovně sdílení, první úroveň se týká sdílení mezi uživatelskou skupinou, data sdílená v druhé úrovni budou veřejná pro všechny registrované uživatele. Pokud nebude nastavena žádná úroveň sdílení, data budou soukromá.

5.1.6 Ukládání výsledků a mezivýsledků výpočtu

Zpracovaná data včetně mezivýsledků budou uložena zpět do databáze aplikace.

5.1.7 Grafický nástroj pro tvorbu a úpravu workflow

Pro vytváření modelů workflow bude vytvořeno přehledné, intuitivní grafické rozhraní. Plátno pro vytváření workflow bude logicky začleněno mezi ostatní funkce portálu, vzhled nástroje bude korespondovat s celkovým designem aplikace.

5.1.8 Správa workflow

Vyhrazená skupina uživatelů bude moci spravovat všechny vytvořené modely v rámci aplikace.

5.1.9 Omezení prostředků pro řízení workflow

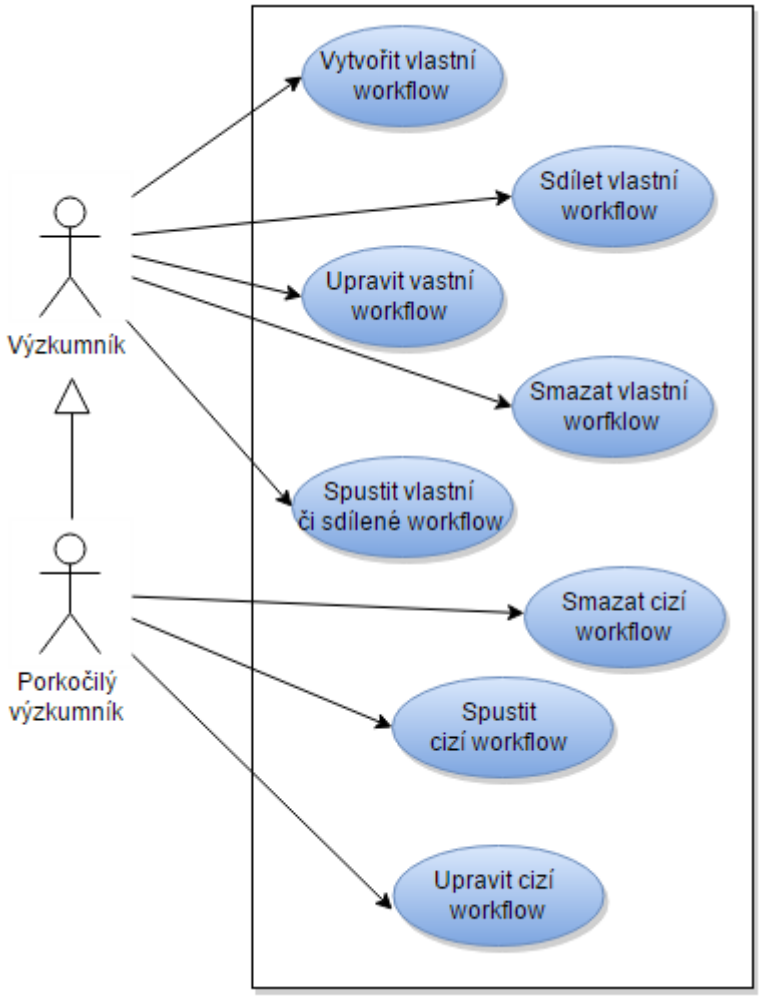
Zpracování jednotlivých workflow nesmí kriticky ovlivnit fungování aplikace. Pro režii týkající se výpočtů workflow bude přidělen konfigurovatelný výpočetní prostor.

5.1.10 Využití otevřených technologií

Vybrané implementační nástroje a knihovny musejí být legálně dostupný open-source.

5.2 Případy užití

Diagram případů užití zobrazuje klíčové vlastnosti systému z pohledu uživatele.



Obr. 11 - Diagram případů užití

5.2.1 Případ 1 – Vytvořit vlastní workflow

Uživatel může v aplikaci poskládat dostupné procedury a vytvořit tak model workflow.

Podmínky pro spuštění:

- Aktér (výzkumník) musí být přihlášen do aplikace.

Základní tok:

1) Součástí uživatelského rozhraní je plátno, na kterém aktér vymodeluje workflow do požadované formy
2) Aktér zvolí možnost uložení modelu
3) Aplikace provede validaci nad zadaným modelem
4) Uživatel je informován o výsledku validace
5) Aplikace uloží model ve validním či nevalidním stavu do databáze. Možnost uložení modelu v nevalidním stavu dává aktérovi šanci se k modelu vrátit a uvést ho do validního stavu později.

Podmínky pro dokončení:

- Model workflow bude uložen do databáze.
- Aktér vidí v aplikaci uložený výsledek.

5.2.2 Případ 2 – Sdílet vlastní workflow

Aktér může sdílet své workflow s ostatními uživateli. Sdílení je možné v rámci skupiny či veřejné sdílení se všemi uživateli EEG/ERP Portálu

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a má uložený nejméně jeden workflow model. Workflow musí být uložené ve validním stavu.

Základní tok:

1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost pro úpravu modelu
2) V aplikaci se na plátně zobrazí vybrané workflow, aktér provede akci pro sdílení workflow
3) Aktér dále vybere viditelnost sdílení (privátní, v rámci skupiny, veřejné)

Podmínky pro dokončení:

- Workflow je v seznamu označeno jako sdílené (v rámci zvolené viditelnosti). Uživatelé ze skupin, pro které bylo sdílení určeno, mají dané workflow viditelné v seznamu sdílených.

5.2.3 Případ 3 – Upravit vlastní workflow

Aktér může sebou vytvořené workflow editovat.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam jeho vlastních workflow obsahuje nejméně jednu položku. Na stavu položky nezáleží.

Základní tok:

1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost upravení položky.
2) V aplikaci se na plátně zobrazí vybrané workflow, aktér modifikuje model dle potřeby.
3) Aplikace provede validaci nad zadaným modelem
4) Uživatel je informován o výsledku validace
5) Aplikace uloží model ve validním či nevalidním stavu do databáze. Možnost uložení modelu v nevalidním stavu dává aktérovi šanci se k modelu vrátit a uvést ho do validního stavu později.

Podmínky pro dokončení:

- Upravený model workflow bude uložen do databáze.
- Aktér vidí v aplikaci uložený výsledek.

5.2.4 Případ 4 – Smazat vlastní workflow

Aktér může sebou vytvořené workflow smazat.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam jeho vlastních workflow obsahuje nejméně jednu položku. Na stavu položky nezáleží.

Základní tok:

1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost smazání položky.
2) Systém provede požadované nastavení.

Podmínky pro dokončení:

- Vybraný model bude odstraněn z databáze a ze všech seznamů, ve kterých dané workflow figurovalo.
- Z databáze budou odstraněna i výstupní data spojená s odstraněným workflow.

5.2.5 Případ 5 – Spustit vlastní workflow

Aktér může dostupná workflow spouštět a pozorovat výsledky.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam jeho vlastních workflow obsahuje nejméně jednu položku ve validním (spustitelném stavu)

Základní tok:

1) Aktér vybere daný validní workflow model ze seznamu a v nabídce zvolí možnost pro spuštění.
2) Aplikace provede validaci nad vybraným modelem
3) Aplikace spustí workflow podle definovaného modelu

Alternativní tok 1:

1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost upravení položky.

2) V aplikaci se na plátně zobrazí vybrané workflow, aktér zvolí akci pro spuštění.
3) Aplikace provede validaci nad vybraným modelem
4) Uživatel je informován o výsledku validace
5) Aplikace spustí workflow podle definovaného modelu
6) Po dokončení aplikace odešle aplikace uživateli email, kde bude oznámení o dokončení výpočtu
7) Aplikace uloží výsledky do databáze

Alternativní tok 2:

1) Pokud validace, která se prováděla před spuštěním, vrátí negativní výsledek (i když byl model uložen jako validní, konfigurace procedur se mohla od doby uložení změnit), uživatel je informován o výsledku validace a požádán o upravení modelu.

Podmínky pro dokončení:

- Uživatel vidí v seznamu workflow, že požadovaná položka je označená jako spuštěná.
- V seznamu je dostupná informace o stavu výpočtu.
- Po dokončení výpočtu dorazí uživateli informační email a v seznamu výsledků jsou uloženy výstupy workflow.
- V případě druhého alternativního toku se nestane žádná akce.

5.2.6 Případ 6 – Smazat cizí workflow

Aktér pokročilý výzkumník může smazat jakékoliv workflow. Tato možnost je k dispozici pro případ nutnosti řešit náhlé problémy týkající se konkrétního modelu.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam workflow obsahuje nejméně jednu položku. Na stavu položky nezáleží.

Základní tok:

1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost smazání položky.
2) Systém provede požadované nastavení.

Podmínky pro dokončení:

- Vybraný model bude odstraněn z databáze a ze všech seznamů, ve kterých dané workflow figurovalo.
- Z databáze budou odstraněna i výstupní data spojená s odstraněným workflow.

5.2.7 Případ 7 – Spustit cizí workflow

Aktér může cizí workflow spouštět a pozorovat výsledky.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam dostupných workflow obsahuje nejméně jednu položku ve validním (spustitelném stavu)

Základní tok, Alternativní tok 1 a Alternativní tok 2:

- Viz případ 5

Podmínky pro dokončení:

- Uživatel vidí ve svém seznamu workflow, že požadovaná položka je označená jako spuštěná.
- Uživatel, který je vlastníkem workflow, není o spuštění informován a nevidí ani žádné výstupy.

- V seznamu workflow je dostupná informace o stavu výpočtu. Po dokončení výpočtu dorazí aktérovi informační email a v seznamu výsledků jsou uloženy výstupy workflow.
- V případě druhého alternativního toku se nestane žádná akce.

5.2.8 Případ 8 – Upravit cizí workflow

Aktér může libovolné workflow editovat.

Podmínky pro spuštění:

- Aktér je přihlášený do aplikace a seznam jeho dostupných workflow obsahuje nejméně jednu položku. Na stavu položky nezáleží.

Základní tok

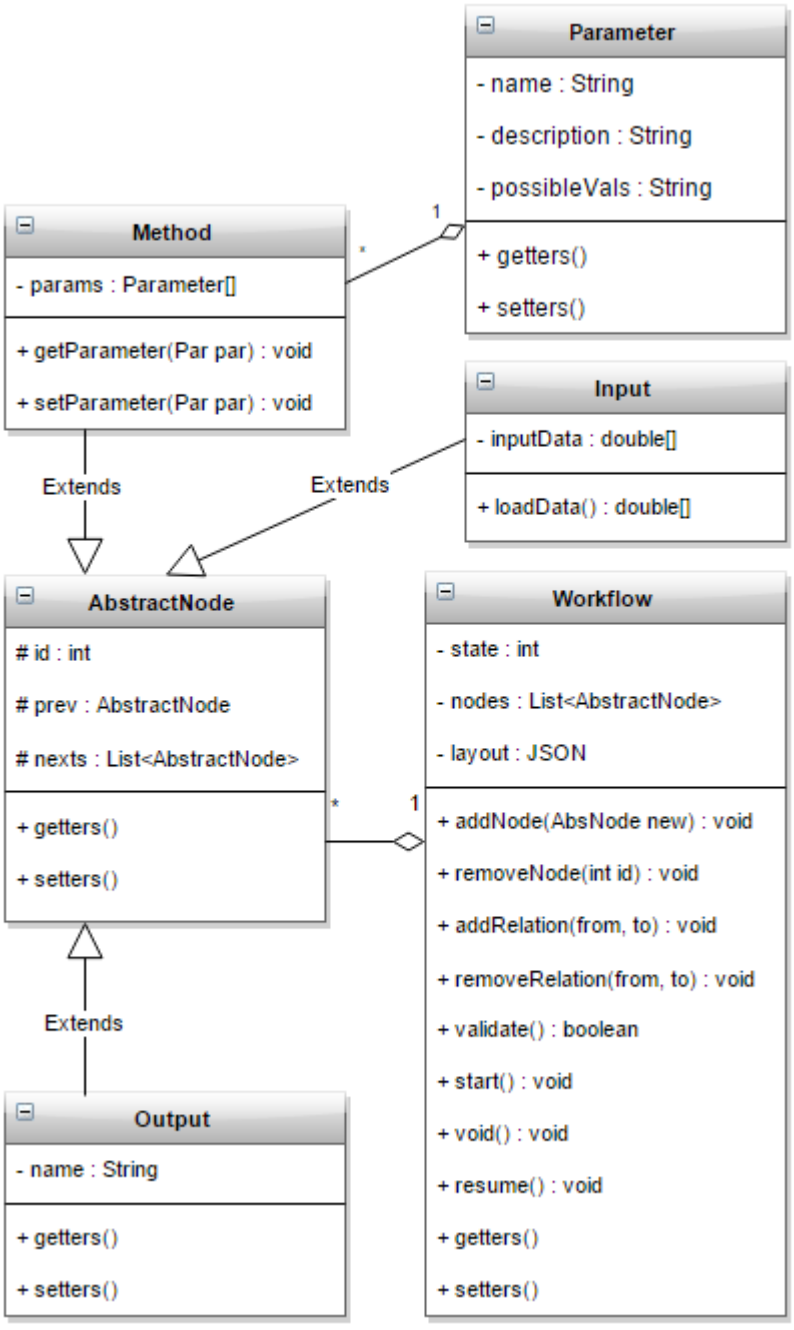
1) Aktér vybere daný workflow model ze seznamu a v nabídce zvolí možnost upravení položky.
2) V aplikaci se na plátně zobrazí vybrané workflow, aktér modifikuje model dle potřeby.
3) Aplikace provede validaci nad zadaným modelem
4) Uživatel je informován o výsledku validace
5) Aplikace uloží model ve validním či nevalidním stavu do databáze. Možnost uložení modelu v nevalidním stavu dává aktérovi šanci se k modelu vrátit a uvést ho do validního stavu později.

Podmínky pro dokončení:

- Upravený model workflow bude uložen do databáze.
- Aktér vidí v seznamu svých workflow uložený výsledek.
- Uživatel, který je vlastníkem workflow, není o akci informován a nevidí ani žádné výstupy.

5.3 Diagram tříd

Model workflow bude definován stromovou strukturou, ve které procedury budou reprezentovány uzly a procedurální návaznosti budou hranami toho stromu. Diagram tříd je zobrazení statické struktury systému prostřednictvím tříd a vztahů mezi nimi.



Obr. 12 - Diagram tříd

5.4 Popis tříd

5.4.1 Třída AbstractNode

Tato třída reprezentuje jeden uzel ve stromovém schématu workflow. Jedná se o uzel nedefinovaného typu, je rodičem všech typů uzlů.

Parametery:

- Id – unikátní číslo v modelu workflow, které jednoznačně identifikuje uzel.
- Prev – odkaz na předcházející uzel stromové struktury workflow
- Nexts – Kolekce (Seznam) uzlů, které na daný prvek logicky navazují.

Metody:

- Třída obsahuje pouze metody pro získání či nastavení privátních atributů.

5.4.2 Třída Parameter

Třída reprezentující jeden vstupní parametr výpočetní procedury.

Parametry:

- Name – název parametru
- Description – krátká vysvětlivka popisující co lze daným parametrem nastavit
- possibleVals – slovní popis možných hodnot či rozsahu intervalu

5.4.3 Třída Method

Třída *Method* reprezentuje výpočetní proceduru pro zpracování dat. Je potomkem třídy *AbstractNode*. Třída bude mít ve struktuře definovaného jednoho předchůdce a minimálně jednoho následníka.

Parametry:

- Params – seznam prvků třídy *Parameter*. Definice všech vstupních parametrů procedury reprezentované třídou *Method*.

Metody:

- Třída obsahuje pouze metody pro získání či nastavení privátních atributů.

5.4.4 Třída Input

Vstupní uzel v modelu workflow. Třída definuje vstupní data.

Parametry

- inputData – pole desetinných čísel (EEG signál)

Metody

- loadData – načte z databáze požadovaný soubor a převede je na vstupní formát.

5.4.5 Třída Output

Třída označující výstupní bod workflow.

Parametry:

- Name – pojmenování výstupního souboru

Metody:

- saveOutput – uložení výstupu do databáze¨

5.4.6 Třída Workflow

Třída reprezentující celý model jednoho workflow. Obsahuje schéma všech uzlů a jejich návaznosti. Třída také uchovává informaci o rozmístění všech prvků v prezentační vrstvě.

Parametry:

- State – stav, ve kterém se dané workflow nachází (validní, nevalidní, spuštěné, dokončené)
- Nodes – seznam všech uzlů v modelu (jednotlivé návaznosti jsou uchovávány v instancích uzlů)
- Layout – informace o rozložení všech prvků na plátně prezentační vrstvy
- UserId – identifikace uživatele, který workflow vytvořil (vlastník workflow)
- SharedState – uchovává informaci o zvolené hladině viditelnosti vůči ostatním uživatelům portálu

Metody:

- addNode, removeNode – přidání či odebrání komponenty (uzlu) do workflow
- addRelation, removeRelation – přidání či odebrání vazby mezi komponentami
- start – tato metoda spustí workflow a předá jej plánovači, pokud je k dispozici volný výpočetní výkon, tak se proces spustí ihned. V opačném případě je úloha na výpočet zařazena do fronty. Logika plánování není v této třídě řešena.

5.5 Analytické procedury jako služby

Protože EEG/ERP Portál nebude provádět vlastní výpočty procedur, ale výpočty budou přenechány aplikaci EEG data processor, je třeba popsat způsob komunikace aplikací.

5.5.1 Webové služby, WSDL, SOAP

Komunikace a výměna dat mezi EEG/ERP Portálem a EEG data procesorem je zajištěna webovou službou. Webová služba je softwarový systém umožňující interakci dvou strojů na síti. Je popsána ve strojově zpracovatelném formátu, konkrétně WSDL. S webovou službou ostatní stroje komunikují způsobem, který je předepsaný v popisu služby, pomocí protokolu SOAP přepravené pomocí jiných, již zavedených protokolů [10].

5.5.2 MTOM

Webové služby nebyly původně navržené pro přenos rozsáhlých dat. Webová služba poskytovaná aplikací EEG data processor využívá mechanismu MTOM (Message Transmission Optimization Mechanism). To je metoda, kterou konsorcium W3C zavadí efektivní posílání binárních dat skrz webové služby. Rozsáhlá data nejsou posílaná uvnitř SOAP obálky, ale jako příloha nebo více příloh vně SOAP obálky. Pokud je odesílána nebo přijímána datová příloha, jsou binární data, ze kterých daná příloha sestává, zadržována odděleně od těla zprávy SOAP, takže nemusí být analyzována jako XML [15]. Výsledkem je efektivnější zpracování, než kdyby byla binární data zadržována v rámci prvku XML.

5.5.3 Apache CXF

V aplikaci EEG data processor je pro vývoj webových služeb použit Framework Apache CXF. Hlavními výhodami frameworku CXF je těsná integrace s frameworkem Spring, lze použít jako Maven plugin, sada anotací pro snadný vývoj a spuštění webových služeb, zabezpečení, snadná konfigurace serveru, automatické generování a publikování WSDL souboru [16].

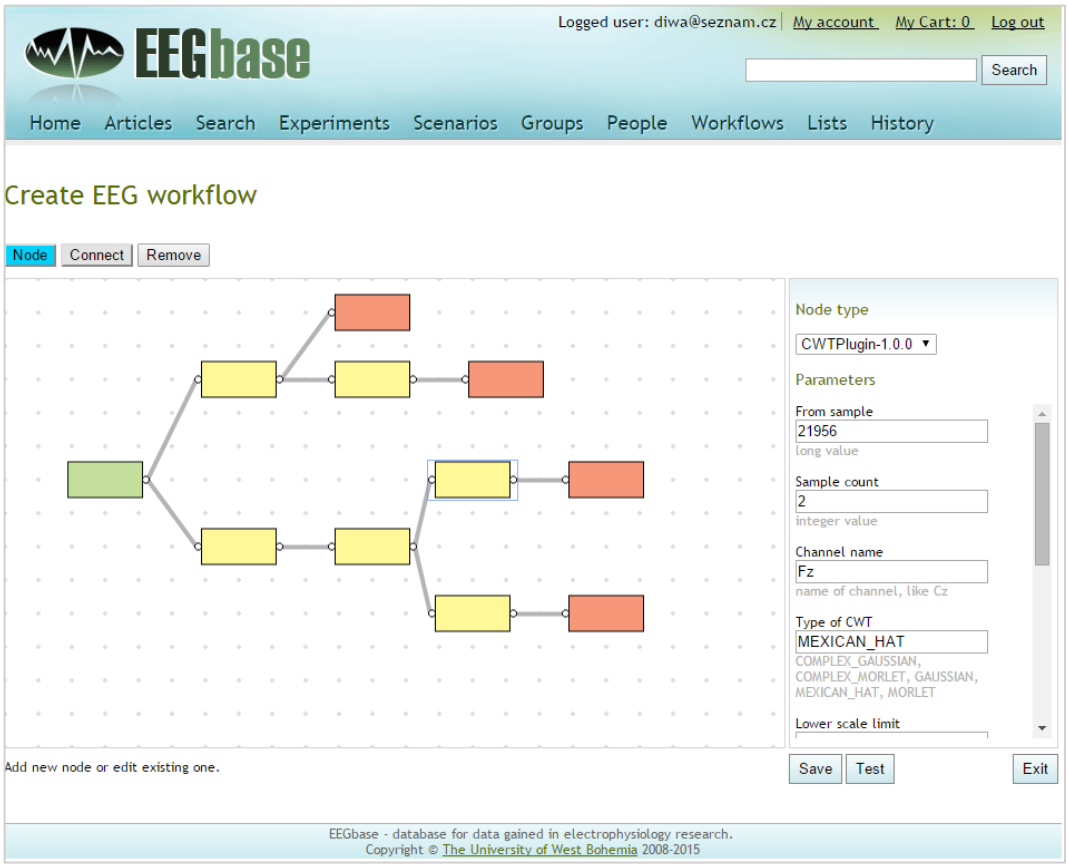
6 Implementace workflow editoru EEG/ERP Portálu

6.1 Prezentační vrstva

Workflow editor byl zakomponován do aplikace, tak aby nenarušil stávající architekturu aplikace a co nejvíce využíval již zavedených technologií.

6.1.1 Webové uživatelské rozhraní

EEG/ERP Portál využívá pro tvorbu uživatelského rozhraní komponentového frameworku Apache Wicket, který zde spolupracuje společně se Spring framworkem, jež tvoří jádro celé aplikace. Do stávající struktury byla přidána komponenta workflow, která má za úkol uživatelům poskytnout přístup ke všem definovaným případům užití.



Obr. 13 - Náhled na editor workflow zabudovaný do EEG/ERP Portálu

6.1.1.1 Wicket

Apache Wicket je framework pro tvorbu webových aplikací v programovacím jazyce Java. Wicket patří mezi komponentně řízené frameworky, tyto frameworky se vyznačují vysokou abstrakcí nad HTTP protokolem. Webová aplikace psaná pod frameworkem Wicket se skládá z prosté HTML šablony pro prezentační vrstvu a Java kódu pro business logiku. Zpracování šablony jako značkovacího jazyka je obecně vzato komplexnější, pro HTML je však podpora většiny komponent včetně AJAXu, Wicket komponenty jsou v HTML souboru označeny speciálním mark-upem [18].

6.1.1.2 Spring

Jádro aplikace EEG/ERP Portál je napsáno pomocí frameworku Spring. Framework je postaven na využití návrhového vzoru IOC (Inversion of Control). Tento návrhový vzor funguje na principu přesunutí zodpovědnosti za vytvoření a provázání objektů z aplikace právě na framework. Objekty lze získat prostřednictvím vsazování závislostí, což je speciální případ Inversion of Control. Dependency Injection řeší vlastní způsob vytvoření a vložení objektů. Objekty vytvořené kontejnerem jsou nazývány Beans. Framework vytvoří Beans na základě načtení konfiguračního souboru ve formátu XML, který obsahuje jejich definice [19].

6.1.1.3 Popis tříd tvořící prezentační vrstvu

- WorkflowsLeftMenu.java – definuje položky menu, které tvoří navigaci workflow sekce. Toto menu je přidané do všech stránek tvořící workflow sekci aplikace. Stránka CreateWorkflowPage je jedinou výjimkou, ve které není menu s navigací zobrazeno z důvodů úspory místa, které je potřeba pro grafický editor workflow.
- EEGProcessingPage – úvodní stránka workflow sekce. Obsahuje textové informace o možnostech tvorby workflow v aplikaci EEG/ERP Portál

- WorkflowDataProvider – potomek Wicket třídy BasicDataProvider, který má za úkol naplnit model tabulek ze stránky WorkflowListPage.
- WorkflowListPage – stránka obsahující dvě tabulky s přehledem dostupných workflow. První tabulka zobrazuje jen workflow vytvořené uživatelem, ve druhé jsou zobrazena všechna dostupná workflow sdílená ostatními uživateli. Data v druhé tabulce jsou seskupena podle úrovně sdílení (v rámci skupiny, veřejná data)
- WorkflowResultsPage – obsahuje seznam výsledků z vykonaných workflow
- CreateWorkflowPage – implementace grafického editoru workflow. Stránka je složena ze tří panelů – EditorPanel, ParameterPanel a Error panel. Třída neimplementuje navigační menu, na této akce je definovaná pouze akce „Back“, která uživatele nasměruje do stránky WorkflowListPage.
- EditorPanel.java – panel obsahující plátno, na kterém lze modelovat workflow. Než je načteno samotné plátno, provádí tato třída inicializaci, během které kontaktuje aplikaci EEG data processor a získává od ní potřebná data. Po dobu inicializace je zobrazena animace symbolizující načítání. Součástí inicializace je i kontrola dostupnosti JavaScriptu. Pokud inicializace selže, je panel nahrazen třídou ErrorPanel
- ParameterPanel.java – při tvorbě modelu workflow je potřeba zadávat parametry jednotlivých komponent workflow. Tento panel zajišťuje dynamické zobrazení příslušných polí pro uživatelem zrovna vybranou komponent.
- ErrorPanel.java – informuje o selhání inicializace plátna.

6.1.2 Technologie grafického rozhraní editoru pro modelování workflow

Pro pohodlné a intuitivní modelování workflow byl vytvořen editor, který využívá technologií HTML5, frameworku Fabric.js a knihovny jQuery.

6.1.2.1 JQuery

JQuery je rychlá, kompaktní a efektivní knihovna pro JavaScript. Usnadňuje práci s doménovým modelem, spoří a zpřehledňuje kód [20]. jQuery je nutná prerekvizita pro používání frameworku Fabric.js

6.1.2.2 Fabric.js

Specifikace HTML5 poskytuje možnost „tvořit" grafický obsah na plátno, které je součástí html dokumentu (html tag canvas). Canvas je dobrým, rychlým řešením pro 2D grafiku, které je podporováno všemi moderními prohlížeči. Nativní API je bohužel velice nízkoúrovňové, pro snazší implementaci a budoucí udržitelnost kódu se nabízí využití HTML5 canvas frameworků [21].

Z frameworků, které jsou poskytovány pod volně šiřitelnou licencí, pro vytvoření workflow editoru nejvíce vyhovuje Fabic.js. Framework Fabric.js poskytuje chybějící objektový model pro práci nad HTML5 plátnem, vrstvu interaktivity, řízení událostí na objekty a celou sadu dalších užitečných nástrojů. Jedná se o plně open-source projekt, který je licencován pod MIT [22].

6.1.3 Implementace grafického rozhraní editoru pro modelování workflow

6.1.3.1 Komponenta - objekt Node

Objektem typu Node jsou na plátně definovány komponenty workflow. Komponentou může být vstup, výstup, procedura či nemusí být definovaná (nová komponenta).

Atributy:

- Poziční atributy – souřadnice x, souřadnice y, výška, šířka
- Atributy fabrik objektu – hasControls : false (objektu nelze měnit velikost, objekt nejde otáčet)
- Vlastnosti komponenty – uživatelem nastavené atributy komponenty
    - Id – identifikace komponenty v rámci workflow
    - nodeType – definuje typ komponenty (vstup, výstup, procedura)
    - inLine – odkaz na vazbu vedoucí na vstup komponenty
    - outLines[] – seznam odkazů na vazby vedoucí z výstupu komponenty
    - params[] – seznam vstupních parametrů potřebných pro spuštění metody

6.1.3.2 Vazba – objekt Line

Objekt Line slouží k propojení jednotlivých komponent, vazba vždy vede z výstupu předchozí komponenty na vstup komponenty následující.

Atributy:

- Poziční atributy:
    - výchozí bod (souřadnice x, souřadnice y)
    - koncový bod (souřadnice x, souřadnice y)
    - strokeLineCap : round – zaoblené konce úseček
- Atributy fabrik objektu:
    - hasControls : false - objektu nelze měnit velikost, objekt nejde otáčet
    - lockMovementX : true – zamknutí vodorovného posunu objektu
    - lockMovementY : true – zamknutí svislého posunu objektu

- perPixelTargetFind : true – objekt je kurzorem detekován dle vykreslených pixelů, nikoliv dle ohraničujícího obdélníku

- Vlastnosti vazby – atributy nutné k uchování a pozdějšímu vyhodnocení modelu
  - nodeFrom – identifikace výchozího uzlu
  - nodeTo – identifikace koncového uzlu

6.1.3.3 Demonstrace vytvoření komponent

Kód pro vytvoření objektu Node:

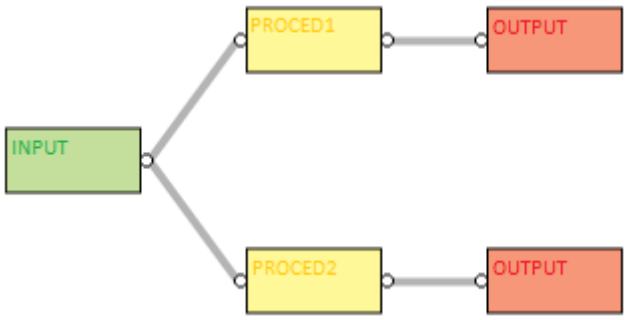
```
function createNode(){
  var pointer = canvas.getPointer(event.e);   //souřadnice kurzoru
  var node = new fabric.Image(undefImage, {
    originX: 'center',
    originY: 'center',
    left: Math.round(pointer.x / grid) * grid,       //uchycení k mřížce
    top: Math.round(pointer.y / grid) * grid,
    width: 80,
    height: 35,
    hasControls: false
  });
  node.nodeType = 0;                    //výchozí typ uzlu
  node.outLines = [];
  node.inLine = 0;
  node.id = objectId;
  objectId++;
  canvas.setActiveObject(rect);

  //překrytí metody pro export objektu
  node.toObject = (function(toObject) {
    return function() {
      return fabric.util.object.extend(toObject.call(this), {
        hasControls: this.hasControls,
        id: this.id,
        nodeType: this.nodeType,
        outLines:  this.outLines,
        inLine: this.inLine
      });
    };
  })(node.toObject);

  canvas.add(node);                     //přidání na plátno
  canvas.moveTo(node, 0);               //posun na popředí
  }
}
```

Kód pro vytvoření objektu Line:

```
function createLine(){
  var points = [ p.getLeft() + 38,
    p.getTop(),
    p.getLeft(),
    p.getTop()  ];
  line = new fabric.Line(points, {
    strokeWidth: 4,
```

```
            lineWidth: 5,
            stroke: 'rgb(180,180,180)',
            originX: 'center',
            originY: 'center',
            strokeLineCap: 'round',
            padding: 20,
            perPixelTargetFind : true,
            lockMovementX : true,
            lockMovementY : true,
            hasControls : false,
            evented : true
           });

           nodeFrom : this.nodeFrom,
           nodeTo : this.nodeTo

           p.outLines.push(this);

           line.toObject = (function(toObject) {
           //překrytí metody pro export objektu
            .
            .
            .
           })(line.toObject);

           canvas.add(line);
           canvas.sendToBack(line);

           isFirstClick = false;
           }
      }
```

Ukázka demonstrující, jak vytvořené komponenty a mezikomponentní vazby vypadají na plátně:

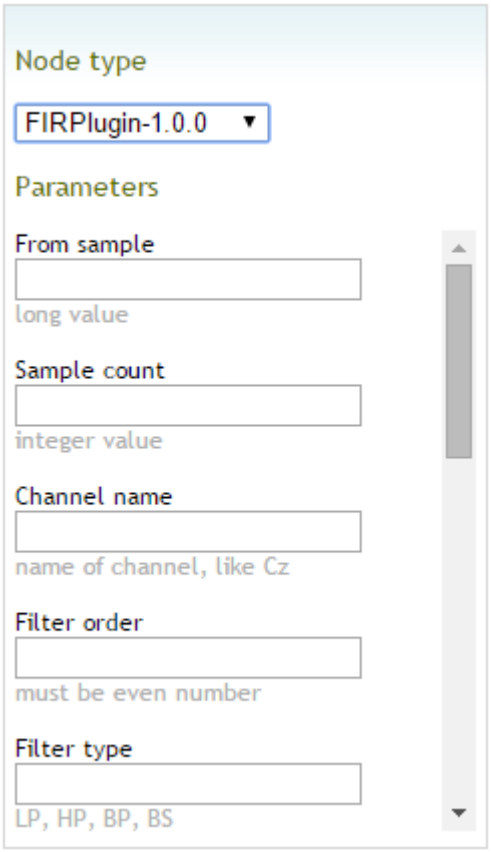


Obr. 14 - Ukázka vytvořeného workflow

6.1.3.4 Volič parametrů komponenty

Pokud jsou na plátně umístěny nějaké komponenty, uživatel je může vybírat a měnit jejich parametry. Pokud se jedná o vstupní komponentu, lze zvolit vstupní soubor. U výstupních komponent je možné definovat název výsledného souboru. U komponent symbolizujících výpočetní procedury

uživatel definuje vstupní parametry, kterými se ovlivňuje průběh výpočtu. Každá výpočetní procedura může mít různý počet parametrů různého typu. Seznam dostupných komponent a definice jejich parametrů jsou při inicializaci plátna načtena z aplikace. Při výběru komponenty se vždy podle jejího typu dynamicky změní všechna pole pro zadání parametrů.



Obr. 15 - Panel pro volbu parametrů komponenty

6.1.3.5 *Informační řádek*

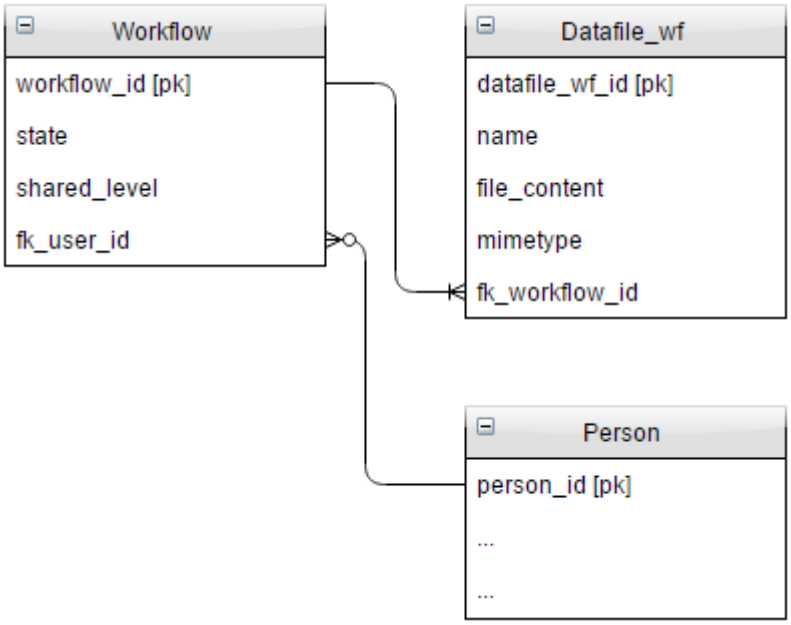
Pod plátnem je umístěn informační řádek – status bar, ve kterém je zobrazena textová nápověda vztahující se k vybrané akci.

6.2 Datová vrstva

6.2.1 Data ukládaná do relační databáze (SQL)

Část informací charakterizující jedno workflow je ukládána do tabulky nazvané Workflow. Jedná se o informace vhodné pro relační model databáze,

hlavně kvůli sloupcům, které jsou tvořeny cizími klíči z jiných tabulek. Dále jsou v relační části umístěna výstupní data, konkrétně se jedná o tabulku *Datafile_workflow*. Pro tyto účely nebylo možné využít stávající tabulku *Datafile*, protože obsahuje nevhodné vazby např. na tabulku *Experiment*. Workflow není s experimentem vůbec svázáno. Tabulka Person v současné databázi již existuje, v diagramu slouží pouze k naznačení vazby.



Obr. 16 - Zjednodušný E-R model týkající se přidaných entit

6.2.1.1 Tabulka Workflow

Nově vytvořená tabulka sloužící k uchování kostry workwlow. Obsahuje následující sloupce:

- Workflow_id – primární klíč, unikátní identifikace objektu.
- State – stav, ve kterém se workflow aktuálně nachází (Připravené, běžící, dokončené)
- Shared_level – flag nesoucí informaci o tom, zda je dané workflow sdílené v rámci skupiny či nikoliv (je privátní)
- User_id – cizí klíč do tabulky Person. Váže workflow k osobě, která ho vytvořila.

6.2.1.2 Tabulka Datafile_wf

Nová tabulka vytvořená pro k uchování výsledků spuštěných workflows. Tabulka má následující strukturu.

- Datafile_wf_id - identifikace datového souboru
- Name – název datového souboru, položka zadaná při vytváření workflow.
- File_content – OID (Object identifier), pointer na binární data
- Mimetype – typ datového souboru, v tomto případě bude ukládána hodnota application/octet-stream

6.2.2 Data ukládaná do nerelační databáze (NoSQL)

Do nerelační části databáze je uložen celý model workflow včetně atributů určující pozici na plátně apod. Framework Fabrik.js umožňuje obsah plátna exportovat do formátu JSON. Tedy stejný formát, jakým databázový systém Elasticsearch reprezentuje indexované položky databáze. Technicky by bylo možné uložit export rovnou do databáze (ajaxový request HTTP PUT přímo z javascriptové funkce), ovšem není to správný postup. Vytvořené workflow musí nejprve projít validací, protože zkušený uživatel může ve svém prohlížeči pomocí vlastního skriptu model libovolně měnit. Správný postup je tedy projít skrze všechny části datové vrstvy (Facade – Service - Dao). Výhodou „odsunutí“ těchto dat do nerelační části je ušetření významného počtu tabulek v relační databázi (Node, Relation, Atributes, Parameter) a to má vliv jak na výkon, tak na přehlednost modelu této databáze.

Ukázka možné podoby exportovaných dat,která budou uložena v nerelační sekci databáze:


```
      {"workflow":[{"type":"line","originX":"center","originY":"center","left"
:630,"top":180.5,"width":44,"height":1,"fill":"rgb(0,0,0)","stroke":"rgb(180,1
80,180)","strokeWidth":4,"strokeDashArray":null,"strokeLineCap":"round","strok
eLineJoin":"miter","strokeMiterLimit":10,"scaleX":1,"scaleY":1,"angle":0,"flip
X":false,"flipY":false,"opacity":1,"shadow":null,"visible":true,"clipTo":null,
"backgroundColor":"","x1":608,"y1":180,"x2":652,"y2":180,"perPixelTargetFind":
```

```
true,"selectable":true,"lockMovementX":true,"lockMovementY":true,"hasControls"
:false,"evented":true,"id":15},{"type":"line","originX":"center
       .
       .
       .
1,"strokeDashArray":null,"strokeLineCap":"butt","strokeLineJoin":"miter","stro
keMiterLimit":10,"scaleX":1,"scaleY":1,"angle":0,"flipX":false,"flipY":false,"
opacity":1,"shadow":null,"visible":true,"clipTo":null,"backgroundColor":"","sr
c":"file:///C:/
","filters":[],"crossOrigin":"","hasControls":false,"id":8,"nodeType":"method2
","outLines":[11],"inLine":10}],"background":""}
```


6.2.3 Fallback NoSQL řešení

NoSQL databáze je v aplikaci zatěžována poměrně málo, pokud by se tedy náhodou diskutovalo o jejím zrušení či nahrazení jinou alternativou, je možné uskutečnit tzv. fallback řešení. Tabulka Workflow se rozšíří o sloupec s názvem JSON, do kterého se uloží exportovaná JSON data jako řetězec. Řešení má ale řadu nevýhod v podobě vlivu na rychlost dotazů (tykající se hlavně fulltextového vyhledávání), nutnost ohlídání tzv. escapování řídících znaků formátu JSON při převodu na řetězec a potřeba zachovat dostatečnou velikostní rezervu (řetězec může být dlouhý).

6.3 Spuštění workflow a management běhu

6.3.1 Přehled tříd

6.3.1.1 WorkflowExecutionManager.java

Obsahuje metody, které od aplikace EEG data processor přes webové služby zjistí jak dostupné analytické metody i s jejich parametry, tak dostupné soubory, které lze vložit jako vstup workflow.

Dále třída před samotným spuštěním parsuje workflow model (převede model, který je v DB uložen jako JSON objekt, na model se strukturou popsanou v kapitole 5.3.) a zvaliduje uživatelem zadané parametry workflow.

6.3.1.2 WorkflowTask

Třída obstarávající režii běhu workflow. Vytvořený task běží ve vlastním vlákně, komunikuje s aplikací EEG data processor, ukládá výsledky workflow.

6.3.2 Task executor

Vytváření instancí třídy WorkflowTask (vytváření nových vláken) je kontrolováno interním plánovačem frameworku Spring, konkrétně typem ThreadPoolTaskExecutor. Plánovač je definován jako Spring bean v souboru applicationContext.xml:

```
<bean id="taskExecutor"
    class="org.springframework.scheduling.concurrent.ThreadPoolTaskExecutor">
    <property name="corePoolSize" value="5" />
    <property name="maxPoolSize" value="10" />
</bean>
```

Parametr "corePoolSize" definuje defaultní počet připravených vláken, v případě potřeby je možno vytvořit falší nová vlákna, jejich maximální počet je definován parametrem "maxPoolSize" (souběžně tedy může běžet až 10 vláken). Pokud ani maximální počet vláken aplikaci nestačí, nová vlákna se ukládají do fronty. Velikost fronty je možné omezit parametrem "queueCapacity". Pokud tento parametr není uvedený, počítá se, jako v tomto případě, s maximálním možnou kapacitou.

7 Ověření správnosti řešení

7.1 Testování funkcionality

7.1.1 Případy užití

Jako základní ověření správnosti implementovaného řešení byly otestovány případy užití podle daného scénáře popsaného v kapitole 5.2. Seznam testovaných případů užití:

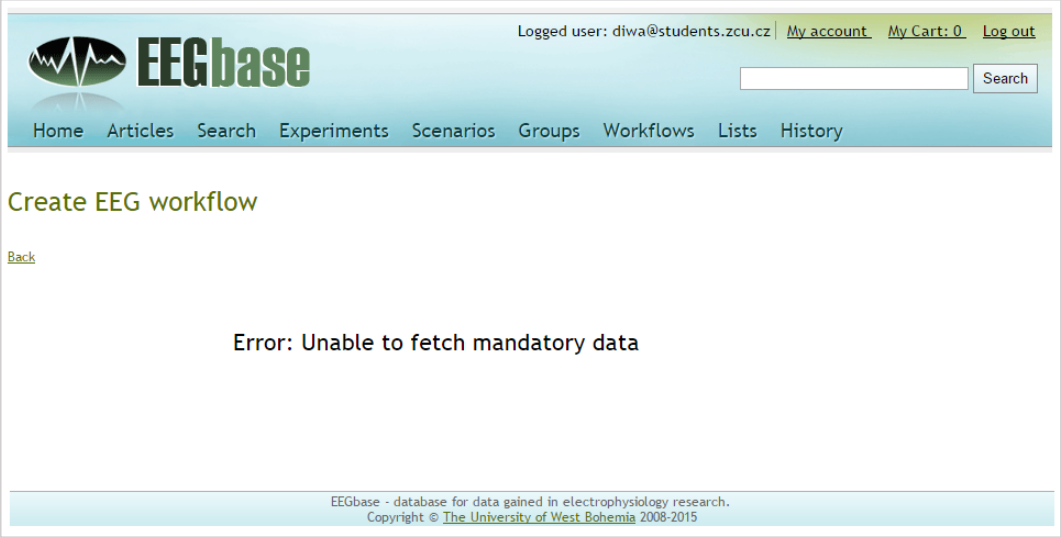
- Případ 1 – Vytvořit vlastní workflow
- Případ 2 – Sdílet vlastní workflow
- Případ 3 – Upravit vlastní workflow
- Případ 4 – Smazat vlastní workflow

- Případ 5 – Spustit vlastní workflow
- Případ 6 – Smazat cizí workflow
- Případ 7 – Spustit cizí workflow
- Případ 8 – Upravit cizí workflow

Výše uvedené případy užití fungovaly podle očekávání a podávaly správné výsledky.

7.1.2 Nedostupnost aplikace EEG data processor

Pokud aplikace EEG data processor nebude dostupná (nekomunikuje pomocí webových služeb s aplikací EEG/ERP Portál), tak aplikace EEG/ERP Portál musí na tento fakt reagovat, žádná uživatelská interakce nesmí skončit běhovou chybou.



Obr. 17 - Chybová hláška o nedostupnosti serveru

Testované úkony v případě nedostupnosti webových služeb:

- Vytvoření nového workflow
- Smazání existujícího workflow
- Editace existujícího workflow

Výše uvedené úkony fungovaly podle očekávání. Při vytváření nového workflow se na pracovním plátně zobrazilo upozornění o nedostupnosti EEG

data processoru. Smazání existujícího workflow proběhlo úspěšně. Při pokusu o editaci existujícího workflow se stejně jako v případě vytváření zobrazilo upozornění.

7.1.3 Spuštění workflow

Uživatel může v průběhu modelování workflow zadat nesprávný parametr dané analytické metody. Workflow s nesprávným parametrem nesmí být spuštěno. Dále musí být před spuštěním všechny bloky workflow spojeny (struktura workflow musí být spojitá). Testované úkony:

- Pole parametru metody je prázdné
- Do pole parametru metody je zadán špatný datový typ
- Workflow je nespojité (chybí minimálně jedna vazba mezi metodami)
- Workflow obsahuje blok, který nemá definovaný typ
- Workflow obsahuje více než jeden vstup
- Workflow obsahuje analytický blok, který nenavazuje na žádný výstup
- Workflow neobsahuje žádný blok

Výše uvedené úkony fungovaly podle očekávání. Před spuštěním daného workflow se ve vyznačeném poli objevila hláška popisující chybu.

7.2 Automatické testování

7.2.1 Framework TestNG

TestNG (Test New Generation) je Framework pro testování Java aplikací, který je inspirovavaný knihovnou JUnit, ale nabízí více funkcí a je snazší na použití [23]. Aplikace EEG/ERP Portál má vysoké procento pokrytí automatickými testy vytvořenými v TestNG.

7.2.2 Vybírání workflow entity z databáze

Třída WorkflowService se stará o správné vybírání objektů z databáze. Testované metody:

- getWorkflowsWhereOwner(Person person);
- getWorkflowsWhereOwner(Person person, int LIMIT);
- getWorkflowsShared(Person owner);
- getWorkflowsShared(Person person, int LIMIT);
- getWorkflowsWhereId(int workflowId);
- int getWorkflowsLastId(Person loggedUser);

Výše uvedené metody fungovaly podle očekávání a úspěšně prošly automatickým testem. O operace vkládání, mazání a editace workflow se stará třída WorkflowFacade, která dané metody dědí od třídy GenericFacade, proto tyto metody nejsou do testů zahrnuty.

7.2.3 Správně nastavená viditelnost seznamů workflow v UI

Uživatel administrátor vidí v seznamu všechna workflow, ostatní uživatelé v seznamu vidí jen svá či sdílená workflow. Tato vlastnost je ověřena automatickým testem.

8 Závěr

Větší část práce byla věnována důkladné analýze zadaného problému. Během fáze analýzy bylo nutné nejprve prostudovat dokumenty týkající se zpracování EEG/ERP signálu a také se seznámit s nástroji pro tvorbu, spuštění a správu workflow. Při zahajování práce přicházela v úvahu myšlenka využití právě nějakého již existujícího workflow systému a jeho následná integrace do aplikace EEG/ERP Portál. Z množiny analyzovaných aplikací se pro tento účel jako nejvhodnější jevil systém Wings. Bohužel při důkladnějším zkoumání byla zjištěna řada jeho nedostatků a neúspěch integrace podpořila sice obsáhlá, ale přesto nedostačující dokumentace projektu. Nenalezení vhodného existujícího nástroje vedlo k příležitosti implementovat vlastní řešení přímo do aplikace EEG/ERP Portál. V této fázi práce byly shromážděny požadavky na systém, začala volba technologií pro implementaci výsledného řešení a následné vytváření prototypů systému. Vhodné řešení bylo využití aplikace EEG data processor, která je podobně jako EEG/ERP Portál vyvíjena výzkumným týmem KIV ZČU. Při implementaci vlastního řešení workflow systému bylo nutné porozumět technologiím použitým v obou aplikacích, tzn. ovládat frameworky Spring, Wicket, Hibernate, TestNG, databázové systémy PostrgeSQL, Oracle a Elasticsearch, build manager Apache Maven a technologie webových služeb Jax-WS, Apache CXF. Při vlastní implementaci prezentační vrstvy pro modelování workflow bylo dále využito technologií Javascript, knihovny jQuery a frameworku Fabric.js, který umožňuje vykreslení grafických prvků do HTML5 komponent.

Implementace zabrala poměrně menší část časového kvanta. Datový model pro ukládání workflow byl úspěšně vytvořen, koresponduje s vytvořeným uživatelským rozhraním pro vytváření, editaci a spouštění workflow. Při implementaci datového modelu byla pro vhodná data využita nerelační databáze. V EEG/ERP Portálu byla vytvořena nová uživatelská sekce

pro správu vytvořených workflow včetně možnosti jejich sdílení s ostatními členy výzkumné skupiny. Do aplikace byla zavedena logika pro spouštění vytvořených workflow. Součástí této logiky je validace zadaných parametrů jednotlivých analytických metod, při chybně zadaných parametrech je uživatel varován dříve, než se workflow spustí. Spotřeba výkonu na proces zpracování workflow je vytvořeným systémem kontrolována a je konfigurovatelná. O vlastní analytické výpočty prováděné při zpracování metod daného workflow se stará aplikace EEG data processor, která byla pro tento účel navržena.

V průběhu práce byla projevena snaha o zavedení sémantiky omezující tvorbu workflow, tak aby vytvořený model workflow byl logicky korektní. Tako omezení nebyla vytvořena, protože sémantický model je příliš komplexní a jeho tvorba je časově i vědomostně náročná.

Výsledek práce rozšiřuje možnosti aplikace EEG/ERP Portál a přidává tak na její atraktivitě a konkurenceschopnosti. Podobně dochází i k lepšímu využití aplikace EEG data processor. Práce pokládá základ možným budoucím rozšířením a vytváří tak prostor pro další zlepšení kvality zpracování elektrofyziologických experimentů na ZČU.

9 Bibliografie

[1] Luck, Steven J. *An Introduction to the Event-Related Potential Technique*. Cambridge: The MIT Press, 2005. [Citace 24. 04. 2015]

[2] BAREŠ, Martin. *Kognitivní evokované potenciály* [online]. [Citace 24. 04. 2015] http://www.csnn.eu/ceska-slovenska-neurologie-clanek/kognitivni-evokovane-potencialy-36052?confirm_rules=1.

[3] ŘONDÍK, Tomáš. *Methods for Detection of ERP Waveforms in BCI Systems.* Plzeň, Rigorózní práce, 2012. [Citace 24. 04. 2015]

[4] Workflow Tutorial. *PNMSoft. [Online]. http://www.pnmsoft.com/contact.* [Citace 25. 04. 2015]

[5] PROCHÁZKA, Jaroslav. *Procesní řízení realizace projektů* [Online]. Ostrava: 2006. [Citace 25. 04. 2015]

[6] A Developer's Introduction to Windows Workflow Foundation (WF) in .NET 4.[Online]. [Citace 25. 04. 2015] http://msdn.microsoft.com/en-us/library/ee342461.aspx //autor + další zdroj obrázku

[7] Windows Workflow Foundation in .NET4. *Somasegar's blog*. [Online]. [Citace 25. 04. 2015] http://blogs.msdn.com/b/somasegar/archive/2010/03/22/windows-workflow-foundation-in-net4.aspx

[87] What is KiSSFLOW?. *KissFLOW.*[online]. [Citace 01. 05. 2015] https://support.kissflow.com/support/solutions/articles/128460-what-is-kissflow-

[9] ŠTĚBETÁK, Jan. *Analytic Methods and Workflows for EEG/ERP Domain.* Plzeň, Rigorózní práce, 2013. [Citace 01. 05. 2015]

[10] System Introduction.*CARMEN.*[online]. [Citace 02. 05. 2015] https://portal.carmen.org.uk/Content/carmen_portalintro.html

[11] About WINGS.*WINGS*.[Online]. [Citace 02. 05. 2015] http://www.wings-workflows.org/about

[12] BYDŽOVSKÝ, Martin. *Relační a nerelační modelování pro portál elektrofyziologických experimentů.* Plzeň, Diplomová práce, 2014. [Citace 03. 05. 2015]

[13] FOWLER, Martin. *NoSQL Distilled*. Addison-Wesley Professional, 2012. ISBN-10: 0321826620. [Citace 03. 05. 2015]

[14] STRAUCH, Christof. *NoSQL DataBases* [online]. [Citace 03. 05. 2015] http://www.christof-strauch.de/nosqldbs.pdf.

[15] Web Services Architecture.*W3C.*[Online]. [Citace 06. 05. 2015] http://www.w3.org/TR/ws-arch/

[16] SOAP Message Transmission Optimization Mechanism.*W3C.* [Online]. [Citace 06. 05. 2015] http://www.w3.org/TR/soap12-mtom/

[17] Apache CXF: An Open-Source Services Framework.*Apache Foundation*.[Online]. [Citace 06. 05. 2015] http://cxf.apache.org/

[18] Wicket feature list.*Apache Foundation.*[Online]. [Citace 09. 05. 2015] https://wicket.apache.org/meet/features.html

[19] Overview of Spring Framework.*docs.spring.io.*[Online]. [Citace 09. 05. 2015] http://docs.spring.io/spring/docs/3.0.x/spring-framework-reference/html/spring-introduction.html

[20] jQuery Introduction.*w3schools*.[Online]. [Citace 09. 05. 2015] http://www.w3schools.com/jquery/jquery_intro.asp

[21] Ten Javascript frameworks. *Backslash.*[Online]. [Citace 09. 05. 2015] http://www.backslash.gr/content/blog/webdevelopment/7-10-javascript-canvas-frameworks

[22] Introduction to Fabric.*Fabric.js.*[Online]. [Citace 09. 05. 2015] http://fabricjs.com/fabric-intro-part-1/#why_fabric

[23] TestNG Overview.TestNG.[Online]. [Citace 13. 05. 2015] http://testng.org/doc/index.html

Příloha A – Vytváření workflows v aplikaci EEG/ERP Portál (uživatelská příručka)

Úvod

Aplikace EEG/ERP Portál umožňuje vytváření workflow sestavené z dostupných analytických metod. Je možné visuálně provádět analýzu naměřených dat. Pro obsluhu workflow je připraveno přehledné uživatelské rozhraní.

Vytvoření workflow a jeho uložení

Postup otevření editoru:

1. Pro vytváření vlastních workflow je nutné být registrován v aplikaci EEG/ERP Potál.
2. Pomocí navigace v hlavním menu aplikace vstupte do sekce „Workflows“
3. V menu sekce zvolte položku „Create workflow“
4. Po připojení k serveru se zobrazí interaktivní plátno pro tvorbu workflow.

Pokud se zobrazila varovná hláška „Error: Unable to fetch mandatory data“, znamená to, že server pro zpracování analytických metod (EEG data processor) není dostupný a akci bude nutné opakovat později.

Přidání nového uzlu workflow

Tlačítkem „New node“ se aplikace přepíná do režimu přidávání a editace uzlů. Uzel se přidá po kliknutí do požadované polohy na plátně. Na pravé straně se v nabídce „Node type“ vybere požadovaný typ uzlu (Vstup, výstup, konkrétní analytická metoda). Pokud je zvolena nějaká metoda, dojde k dynamickému

rozbalení polí pro zadání vstupních parametrů. V případě, kdy je zvolen vstup, vybere se jeden z nabízených vstupních souborů. Jediným parametrem výstupního uzlu je název výstupního souboru.

Přidání vazby mezi uzly

Po stisknutí tlačítka „Connect“ se aplikace přepne do režimu přidávání vazeb mezi jednotlivými uzly. Přidání vazby se provede kliknutím na výchozí uzel a následným tažením linky do požadovaného uzlu.

Kliknutím na tlačítko „Save“ dojde k výzvě pro zadání jména vstupního souboru, teprve poté dojde k uložení workflow a následnému přesměrování na přehled uložených workflows.

Přehled workflow a akce nad nimi

V sekci „My workflow“ je zobrazen přehled všech uložených workflows. Dostupné akce jsou zobrazení daného workflow (Klik na odkaz „View“) a smazání vybraného workflow (klik na odkaz „Delete“). Před smazáním workflow je uživatel pro kontrolu dotázán na potvrzení své akce.

Spuštění workflow

Spouštění vybraného workflow probíhá pomocí tlačítka „Run“. Nejprve v aplikaci dojde k validaci vstupních dat, poté začne vlastní výpočet a uživatel je přesměrován do sekce, kde je přehled výstupních souborů. Pokud během validace dojde k chybě, je uživatel upozorněn o charakteru chyby rudým písmem v oblasti nad editorem. Nejčastěji se jedná o nevložený vstupní soubor či chybný datový typ metody. Po odstranění závady se akce „Run musí zopakovat“.