Shannonova jazyková hra

Motto: Tradiční způsob sběr vstupních dat využitelných v rámci výpočetní lingvistiky je zdlouhavý a náročný na finanční prostředky. Zkusme tedy sbírat tato vstupní data prostřednictvím hry, které se bude moci zúčastnit kdokoliv.

poslední úprava: 12.06.2008 02:21

Obsah

1. Motivace a cíle projektu

Motivací projektu je snaha o snížení pracnosti a finančních a časových nákladů na sběr dat využitelných ve výpočetní lingvistice prostřednictvím zajímavé a atraktivní hry dostupné široké skupině uživatelů Internetu. Jako inspirace může posloužit například projekt http://www.espgame.org/ který prostřednictvím jednoduché hry sbírá data o klasifikaci obrázků.

Mezi hlavní cíle první fáze projektu patří zejména

- sběr obecných požadavků
- výběr vhodných technických prostředků
- návrh architektury aplikace
- návrh zjednodušené architektury prototypu
- implementace prototypu s jednou hrou
- ověření využitelnosti dat získaných dat

2. Účastníci

Následuje seznam hlavních osob účastnících se projektu, kontaktní informace a jejich hlavní kompetence:

- **Barbora Hladká**
  - řízení projektu na straně MFF UK, dohled nad odbornou stránkou projektu
  - e-mail: hladka@ckl.mff.cuni.cz
- **Kiril Ribarov**
  - zadavatel projektu, dohled nad odbornou stránkou projektu
  - e-mail: ribarov@ckl.mff.cuni.cz
  - tel.: (+420) 777574227
- **Tomáš Vondra**
  - za návrh architektury, implementace ne-flashových částí aplikace, apod.
  - e-mail: tv@fuzzy.cz
  - tel.: (+420) 732358545
- **David Šťastný**
  - návrh a implementace grafické části aplikace (Flash, HTML, CSS)
  - e-mail: david.stastny@tritonus-tech.com
  - tel.: (+420) 737540379

3. Odkazy na projektové zdroje

- SVN repository: http://svn.fuzzy.cz/shannon
- Project management: http://achievo.fuzzy.cz
- Bug tracker: http://mantis.fuzzy.cz

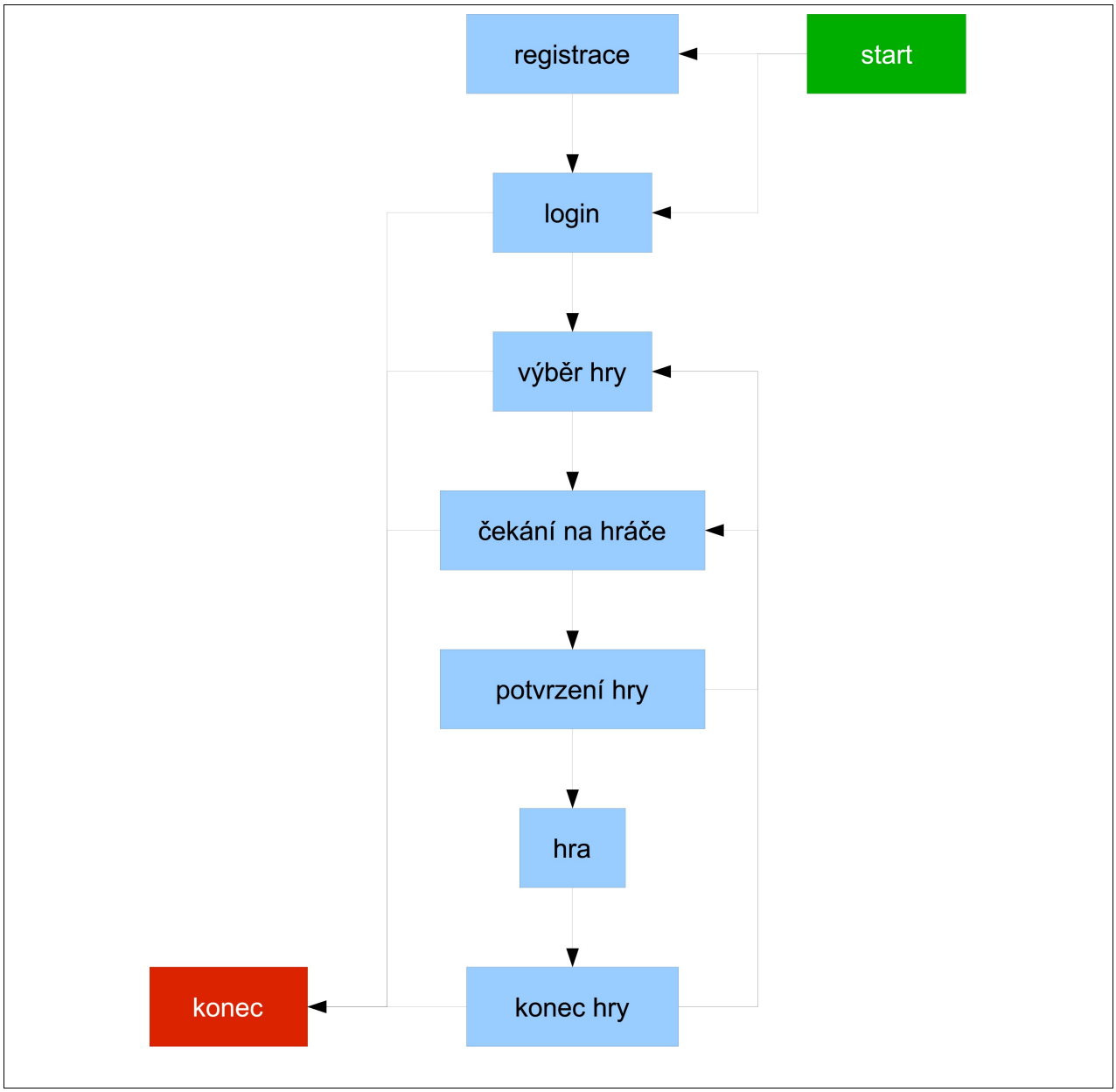
4. Popis implementované hry

Tato část specifikace obsahuje popis algoritmu a základních pravidel implementované hry - uvedená pravidla jsou pouze návrhem, a jejich úprava je v případě potřeby možná.

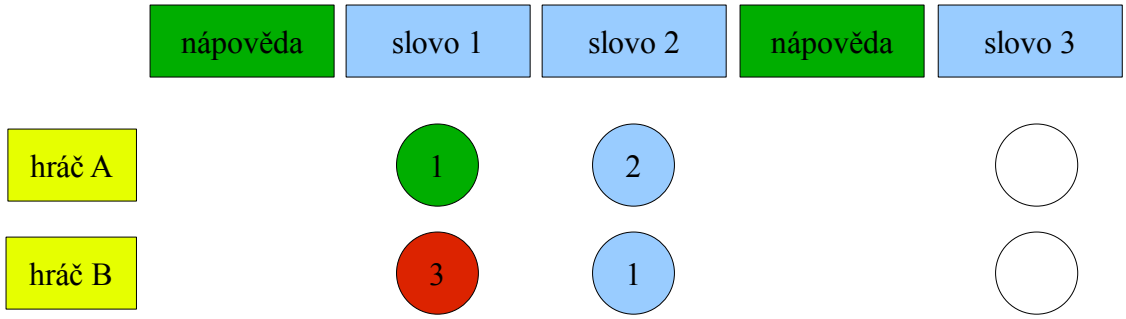
4.1. Základní algoritmus hry

1. Uživatel se zaregistruje (např. prostřednictvím k tomu určeného formuláře) a přihlásí se na herní portál, zvolí hru kterou chce hrát a požadovanou obtížnost - tím se z něj stává potenciální hráč a postupuje do kroku 2 a čeká na protihráče.
2. V okamžiku kdy se na serveru objeví vhodný volný hráč je uživateli nabídnuto zahájení nebo zrušení partie. V případě že oba hráči se zahájením partie souhlasí, začíná hra a oba postupují do kroku 3. V případě že jeden z hráčů se zahájením partie nesouhlasí, čekají hráči dále.

   Velmi důležitou roli hraje anonymita - hráči nesmí mít možnost předem zjistit kdo je jejich soupeř. Při samotném "párování" uživatelů je vhodné brát v potaz i takové informace jako je například použité grafické rozhraní (v případě že jich daná hra má několik) apod.
3. Po zahájení partie se hráčům zobrazí neúplná věta a jejich úkolem je doplnit chybějící slova. Přitom na každé slovo mají omezený počet pokusů, a na každý pokus mají omezený časový interval. Počet vyplněných slov, počet pokusů a délka intervalu na jeden pokus závisí na zvolené obtížnosti hry.
4. Hráči postupně hádají jednotlivá slova věty (mohou hádat i na přeskáčku) a snaží se slova uhádnout na co nejméně pokusů a v co nejkratším časovém intervalu. Během hry se uživatelům zobrazují informace o postupu hry u protihráčů (uhádnutá / neuhádnutá slova, počet pokusů, atd.)
5. Po skončení partie na obou stranách (uhádnutí všech slov / vyčerpání všech pokusů / uběhnutí časového limitu) je hráčům prozrazena správná věta, zobrazeny podrobnosti o hře protihráče (správné a chybné odpovědi, apod.) a celkové vyhodnocení hry (bodové ohodnocení, atd.).



4.2. Schematický návrh grafického rozhraní hry



Uvedený návrh grafického rozhraní obsahuje následující elementy:

- hádanou větu, skladající se z
    - nápověd (slova známá na začátku partie)
    - slov která mají být během partie uhádnuta
- terče pro hráče účastnící se dané partie, znázornující "uhádnutost" daného slova, tj.
    - počet využitých pokusů (0 až 3)
    - úspěšnost (uhádnuto, neuhádnuto ale ještě má alespoň jeden pokus, neuhádnuto a žádné další pokusy už nemá)

Je teoreticky možné že jedna hra bude mít více grafických variant - v tom případě je vhodné tyto varianty držet buď jako zcela samostatné hry, nebo alespoň uchovávat informaci o použitém grafickém rozhraní u dané partie (a hráče, pokud se v jedné partii mohou kombinovat různá grafická rozhraní) - tato informace je důležitá kvůli porovnání dat z různých grafických variant hry (Např. co když se v jedné variantě dá slovo zadat rychleji než ve druhé?)

4.3. Bodové ohodnocení hry

Bodové ohodnocení musí být zvoleno tak aby pokud možno

- odměňovalo rychle a správně hádající uživatele
- penalizovalo uživatele hádající pomalu a nebo s velkým počtem chyb
- zohledňovalo složitost partie (počet nápověd, počet hádaných slov, apod.)
- bylo odolné vůči automatickému hádání

Předběžný návrh bodového ohodnocení je následující:

- bodové ohodnocení celé partie je rovné průměrnému bodového zisku za hádaná slova
- uhádnutí slova v 1. pokusu: 40 bodů
- uhádnutí slova ve 2. pokusu: 20 bodů
- uhádnutí slova ve 3. pokusu: 10 bodů
- nedodržení časového limitu nebo neuhádnutí slova: -10 bodů
- maximální možný zisk na jedné partii tak může být 40 bodů, minimální -10 bodů (ztráta)

Bodová ohodnocení z různých her a nebo obtížností se nekombinují. Konkrétní hodnoty bodového ohodnocení jsou pouze orientační, a je možné je během vývoje poupravit.

4.4. Hra více než dvou hráčů

Je možné / pravděpodobné že v blízké budoucnosti bude vhodné hru modifikovat tak aby ji mohlo hrát i více než dva hráči. Přizpůsobení grafického rozhraní navrhovaného v jednom z předchozích odstavců je přímočaré - stačí jednoduše zobrazit řádky pro další hráče.

Možnost tohoto rozšíření by proto měla brát v potaz již prototypová implementace, tj. měla by ho přímo podporovat (například prostřednictvím k tomu určeného konfiguračního parametru) nebo jeho implementaci v maximální možné míře ulehčit. Tuto možnost je tedy třeba zohlednit při návrhu databázového schématu a komunikace mezi serverovou a Flashovou částí prototypu.

4.5. Sledovaná a ukládaná data o partiích

U každé partie jsou sledována následující informace

- jazyk
- hádaná věta (resp. odkaz na hádanou větu)
- seznam hráčů
- nápověda (předvyplněná slova, slovní druhy neznámých slov, apod.)
- parametry složitosti (počet pokusů na slovo, časový limit)

Nejzajímavějšími daty o partiích však nejsou konečné výsledky ale právě informace o jejím průběhu, tj. o jednotlivých pokusech. U každého pokusu jsou sledovány následující informace

- zadané slovo
- číslo pokusu u tohoto slova (1, 2, 3, ...)
- čas odeslání (vzhledem k začátku partie, případně k předchozímu pokusu nebo slovu)
- výsledek (správně / špatně)

5. Požadavky na funkčnost

V této kapitole je stručně popsány požadavky na funkčnost aplikace (resp. jejího prototypu), společně se stručným popisem technické realizace datné funkčnosti (případně více alternativ řešení).
Požadavky na funkčnost lze zhruba rozdělit do následujících skupin, popsaných samostatně v dalších odstavcích

- správa uživatelů
- správa nainstalovaných her
- řízení průběhu hry
- export sesbíraných dat

5.1. Správa uživatelů

- registrace a přihlášení uživatele (o každém uživateli je uchovávána informace o věku, pohlaví a rodném jazyce, e-mail a login)
- změna informací o uživateli
- přihlášení a odhlášení uživatele
- zobrazení informací o získaných bodech (pro jednotlivé hry)
- zaslání zapomenutého hesla

5.2. Správa nainstalovaných her

- stažení hry
- instalace nové hry

5.3. Řízení průběhu hry

- zobrazení aktuálně dostupných (nainstalovaných) her
- výběr hry a její spuštění

- řízení průběhu hry (vyhodnocování dat z Flashe, předání odpovědí)
- vyhodnocení výsledků hry

5.4. Import vstupních dat

Vstupními daty pro českou variantu hry je XML soubor ve formátu “csts” obsahující data z Českého Národního Korpusu (ČNK). Importovány budou pouze relevantní informace, tj. zejména

- tokenizace (rozdělení na věty a slova)
- vybrané morfologické informace (slovní druh, ...)

Je vhodné filtrovat pouze věty vhodné délky, tj. ani příliš krátké, ani příliš dlouhé (např. délka 10 zní rozumně). Stejně tak je ale možné importovat všechny věty a délku věty používat jako kritérium složitosti partie.

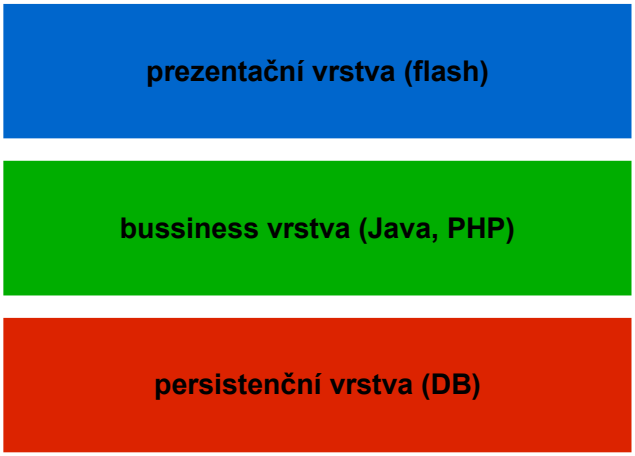
5.5. Export sesbíraných dat

- Během vývoje prototypu je třeba upřesnit které informace jsou pro další zpracování nutné a relevantní, a v jakém formátu jsou tato data očekávána - obojí bude dáno zejména aplikací která bude využita pro zpracování získaných dat.

6. Navrhovaná architektura

S ohledem na relativní jednoduchost technické stránky celého problému navrhujeme využít klasickou třívrstvou architekturu, tj. rozdělit systém do následkujících vrstev (podrobněji popsaných níže):

- **prezentační** - reprezentována hrou ve formě flashové animace, běží v browseru
- **business** - hlavní část funkčnosti (přijímá a vyhodnocuje data z prezentační vrstvy, zasílá odpovědi flashi, výsledky her ukládá do databáze, atd.), běží na serveru
- **persistenční** - zodpovědná jen a pouze za ukládání / načítání dat z databáze



6.1. Persistenční vrstva

Persistenční vrstva neobsahuje žádnou aplikační logiku jelikož je zodpovědná za ukládání a načítání dat (entit popsaných dále) z relační databáze, a efektivně tak odstiňuje tak vyšší vrstvy od specifik datového modelu.

Tato vrstva je realizovánía RDBMS systémem (pro dobré zkušenosti navrhujeme PostgreSQL 8.2.x)

s vytvořeným datovým modelem, a tenkou vrstvou procedur / objektů které zajišťují přístup z business vrstvy, realizované například v PHP (v případě prototypu) resp. Javy (v případě finální aplikace, pokud se PHP během prototypu ukáže jako nepostačující).

6.2. Business vrstva

V této vrstvě je implementována v podstatě veškerá aplikační logika, počítaje autentizací uživatele, přes započetí a řízení průběhu hry, až po vyhodnocování výsledků.

Tato vrstva musí být implementována ve stejném jazyce jako vrstva persistenční, tj. PHP v případě prototypu a případně Javu pro finální aplikaci.

6.3. Prezentační vrstva

Prezentační vrstva je realizována flashovou animací, a jako taková běží v prohlížeči na straně klienta. Vzhledem k bezpečnostním rizikům bude veškerá aplikační logika implementována na straně serveru, tak aby se do flashové animace nedostala citlivá data (viz. dále), a je vykonávána na základě komunikace mezi animací a business vrstvou.

Mezi citlivá data, kterí se do prezentační vrstvy (resp. na klientskou část) nesmí dostat, jsou např.:

- správné odpovědi - analýzou/dekompilací flashové animace nesmí být možné zjistit správnou odpověď, atd.
- informace o druhém hráči - hráči o sobě nesmí vědět aby nemohli spolupracovat například za účelem zvýšení skóre

6.4. Komunikace mezi prezentační a business vrstvou

Komunikace mezi prezentační vrstvou (Flash) a serverovou částí (PHP skripty v prototypu) probíhá prostřednictvím standardních POST požadavků, přičemž komunikace je vždy iniciována ze strany Flashe (Flash zasílá request, serverová část response).

Konkrétní struktura požadavků (zejména předávané proměnné atd.) bude definována až během tvorby prototypu, nicméně měla by být dostatečně flexibilní aby umožnila například účast více než dvou hráčů na hře, a podobně.

7. Rozšiřitelnost

Při návrhu architektury je třeba uvažovat také o možných budoucích rozšířeních aplikace. Přesný rozsah vyžadovaných změn lze předpovídat jen obtížně, nicméně změny lze zhruba rozdělit do dvou hlavních skupin - změny při úpravě existující hry, a úpravy nutné při vytvoření a spuštění nové hry.

7.1. Změny v existující hře

Změny v existující hře mohou mít různý rozsah. Týkají-li se pouze grafické podoby hry, jsou proveditelné pouze změnou flashové animace a zbytku aplikace se nedotkou. Týkají-li se však například rozšíření množiny sbíraných dat nebo logiky hry, je nutná změna datového modelu a pochopitelně také business vrstvy.

V případě netriviálních změn hry (tj. v podstatě čehokoliv kromě triviálních grafických změn) lze za vhodnější považovat vytvoření modifikace jako zcela nové hry - předejde se tím problémům v konzistenci dat. Například:

- po přidání nové informace mezi zaznamenávaná data tato informace ve starých datech chybí
    - Jak se s tímto stavem vyrovnají metody pro analýzu dat?

- Jak se na persistenční vrstvě jednoduše zajistí konzistence dat?

- po rozsáhlejší úpravě grafického rozhraní se může výrazně snížit nebo naopak zvýšit čas potřebný na zadání odpovědi
  - Jak se s tímto stavem vyrovnají metody pro analýzu dat?
  - Jak do dat zaznamenat která data byla získána ze staré a která z nové verze hry?

7.2. Vytvoření a spuštění nové hry

Vzhledem k nepředvídatelným rozdílům mohou rozdíly v povaze her je velmi pravděpodobné, že při vytváření nové hry budou nutné zásahy ve všech třech vrstvách aplikace, jejichž stručný přehled je uveden zde:

- **persistenční vrstva**
  - vytvoření tabulek v datovém modelu - nelze předpokládat že by data pro novou hru bylo možné efektivně a spolehlivě ukládat do struktur optimalizovaných pro jiné hry
  - vytvoření persistenční vrstvy nad těmito novými tabulkami
  - úprava exportu dat
- **business vrstva**
  - implementace kontroly odpovědí, atd.
  - komunikace s flashovou animací
- **prezentační vrstva**
  - nové uživatelské rozhraní

8. Bezpečnost

Při návrhu architektury a následné implementaci aplikace musí být brána v potaz možná bezpečnostní rizika, jako jsou například pokusy o podvody při bodování (například s ohledem na uvažované odměny nejlepším uživatelům), pokusy o nasazení robotů, apod.

9. Entity vyskytující se v aplikaci

- **uživatel** - zaregistrovaný uživatel (relace “players”)

Název	**Datový typ**	**Popis**
id	integer	interní identifikátor uživatele (PK)
email	varchar(64)	e-mailová adresa uživatele
login	varchar(16)	unikátní jméno uživatele, pouze alfanumerické znaky)
password	varchar(32)	heslo (hashované algoritmem MD5)
language	char(2)	rodný jazyk uživatele
birth_year	integer	rok narození

- **hra** - reprezentuje hru zavedenou do systému (relace “games”)

Název	**Datový typ**	**Popis**

id	integer	interní identifikátor hry (PK)
name	varchar(16)	název hry
description	text	popis hry
active	boolean	je hra aktivní (zobrazována)
url	varchar(255)	URL kde je umístěna flashová hra
image	bytea	data obrázku

- **sentence** - věta (relace “sentences”)

Název	**Datový typ**	**Popis**
id	integer	interní identifikátor věty (PK)
tokens	integer	počet tokenů ve větě (slova + interpunkce)
words	integer	počet slov ve větě
language	char(2)	jazyk věty

- **token** - tokeny ze kterých se věta skládá (relace “tokens”)

Název	**Datový typ**	**Popis**
sentence_id	integer	odkaz na hru (FK, PK)
position	integer	pozice slova ve větě (PK)
token	varchar(16)	slovo (resp. token)
type	char(1)	W - slovo, P - interpunkce
flags	varchar(16)	morfologické značky slova (zatím se nepoužívá)

- **match** - reprezentuje realizaci hry mezi dvěma (nebo více) uživateli (relace “matches”)

Název	**Datový typ**	**Popis**
id	integer	interní identifikátor zápasu (PK)
game_id	integer	odkaz na hru (FK)
sentence_id	integer	odkaz na větu hádanou v této hře (FK)
started	datetime	datum a čas začátku hry
finished	datetime	datum a čas konce hry
players	integer	počet hráčů ve hře (>= 2)
level	integer	stupeň obtížnosti (1 - nejlehčí, 3 - nejtěžší)

- **pokus** – krok ve hře (hádání slova), je součástí zápasu (relace “guesses”)

Název	**Datový typ**	**Popis**
match_id	integer	odkaz na hru (FK)
player_id	integer	odkaz na hráče (FK)

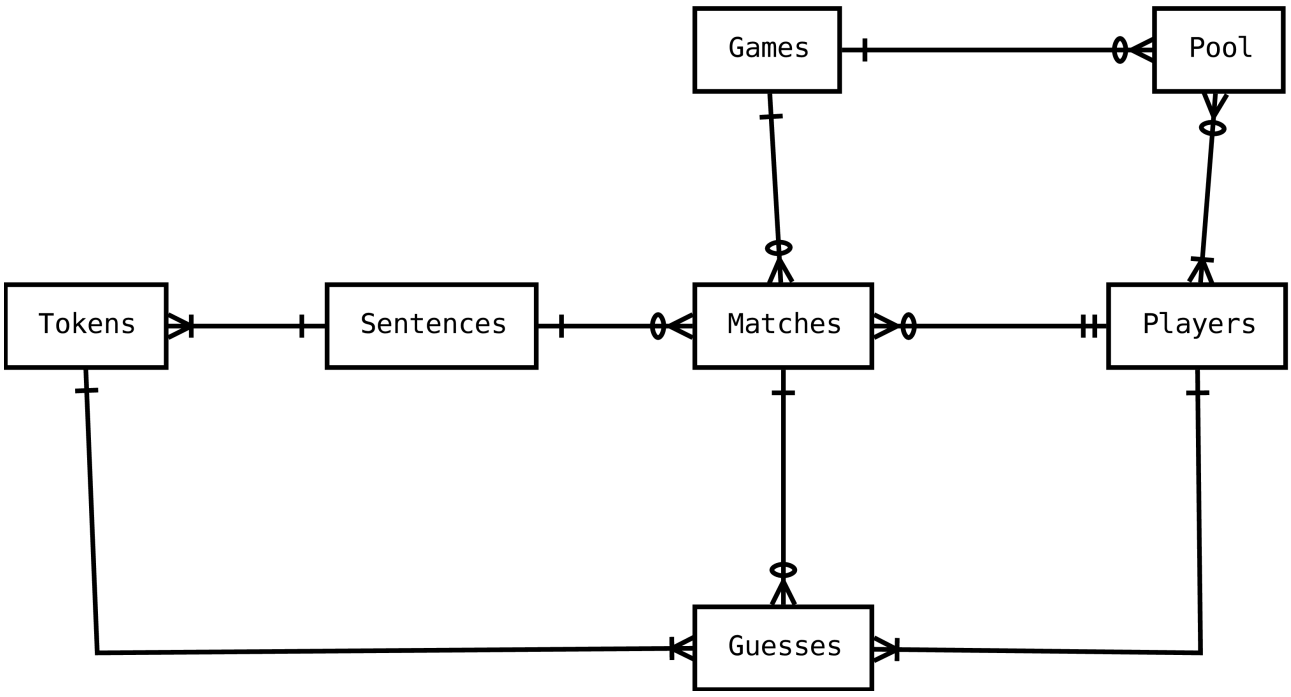
sentence_id	integer	odkaz na větu (FK)
position	integer	pozice tokenu ve větě (společně se sentence_id odkaz na token (FK))
attempt	integer	pořadí pokusu pro tento token
created	datetime	datum a čas pokusu
guess	varchar(32)	hádané slovo

- **pool** - hráč čekající na protihráče (relace “pool”)

Název	**Datový typ**	**Popis**
game_id	integer	odkaz na hru (FK)
player_id	integer	odkaz na hráče (FK)
last_update	datetime	datum / čas posledního přístupu do poolu
level	integer	požadovaná obtížnost hry (1 - 3)
language	char(2)	požadovaný jazyk hry

9.1. Zjednodušené schéma

V následujícím schématu nejsou znázorněny vazební tabulky.



10. Prototypová aplikace

10.1. Zjednodušení v prototypu

Prototypová aplikace neobsahuje:

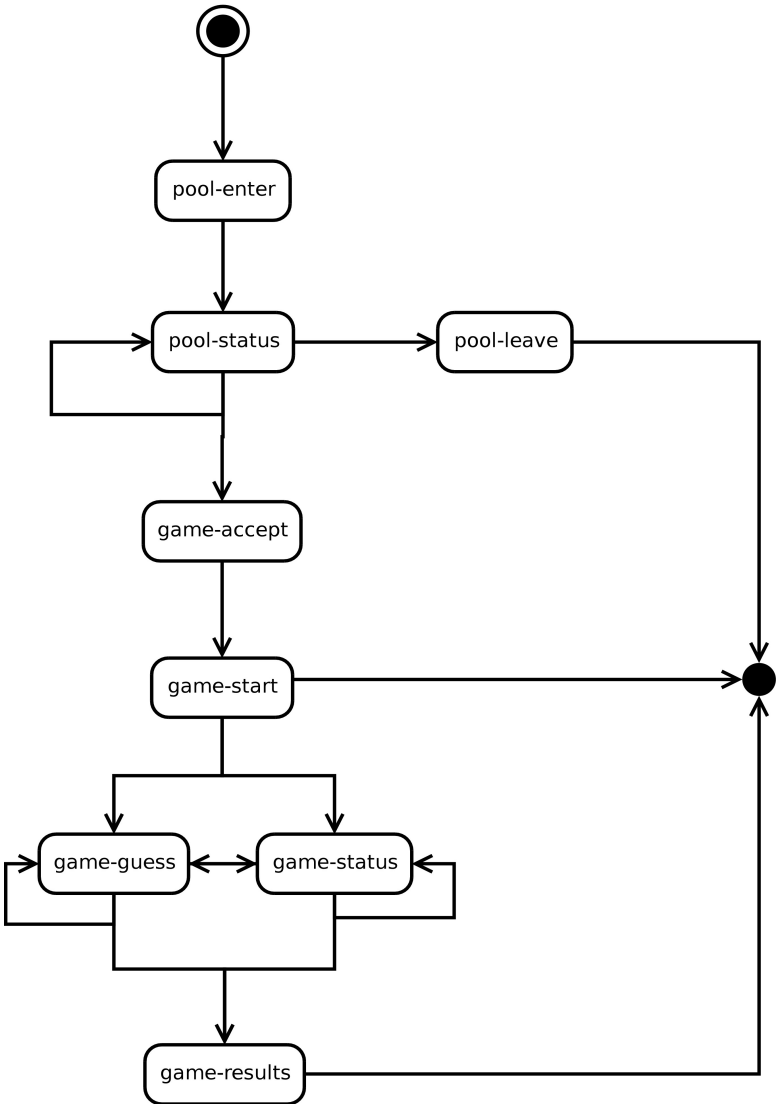
- sofistikovanou implementaci exportu dat - export je řešen jednoduchým skriptem
- možnost automatické instalace a odinstalace hry

10.2. Komunikace mezi Flashovou a PHP vrstvou

Pro fungování prototypu jsou třeba následující "zprávy" zasílané mezi Flashovou a PHP vrstvou:

- pool-enter
- pool-status
- pool-leave
- game-accept
- game-start
- game-status
- game-guess
- game-results

Některé informace (např. ID uživatele) jsou načítány z PHP session, a v následujících odstavcích je to u nich uvedeno jako poznámka. Zbývající parametry jsou načítány z URL (GET požadavek).



10.2.1. pool-enter

Tuto zprávu Flash vysílá pokud uživatel chce vstoupit do poolu a čekat na protihráče v dané hře. Před skutečným vložením do poolu je zkontrolováno zda už v poolu náhodou nečeká potřebný počet protihráčů. Pokud jsou vhodní protihráči nalezeni, je rovnou vytvořena hra a Flash přechází na "game-accept". Jinak je uživatel vložen do poolu a pokračuje na "pool-status".

Při vytváření hry je náhodně vybrána vhodná věta, tj. věta v požadovaném jazyce a odpovídající dané obtížnosti (v úvahu je brán počet slov, zvolený počet nápověd, apod.)

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **game ID** - identifikace požadované hry
- **level** - požadovaná obtížnost hry (1 - lehká, 2 - normální, 3 - obtížná)
- **language** - požadovaný jazyk hry

Návratové hodnoty:

- **status = ENTERED** - protihráči nebyli nalezeni a uživatel vstoupil do poolu
- **status = MATCH_CREATED** - protihráči byli nalezeni, a byla vytvořena hra
- **status = ERROR** - došlo k chybě při zpracování
- **time = "unix timestamp"** - čas odpovědi (standardní UNIX timestamp)

Navazující zprávy:

- **pool-status** - vhodní protihráči nebyli nalezeni, a uživatel tedy vstoupí do poolu
- **game-accept** - vhodní protihráči byli nalezeni, a hra je tak vytvořena automaticky (bez skutečného vstupu do poolu)

10.2.2. pool-status

Pokud uživatel vstoupil do poolu, dotazuje se na aktuální stav, tj. zejména zda nebyli nalezeni vhodní hráči a nebyla vytvořena hra. Pokud je uživatel (pro danou hru) nalezen v poolu znamená to že žádní vhodní protihráči zatím nebyli nalezeni, a je mu vrácen stav "WAITING" a Flash by se po chvíli měl ptát znovu. Pokud hráč (resp. kombinace hráš + hra) v poolu nalezen není, je zkontrolováno zda neexistuje nově vytvořená hra čekající na potvrzení (odmítnutí) - v tom případě je uživateli vrácen stav "MATCH_CREATED" a Flash pokračuje na "game-accept". Pokud hráč není nalezen v poolu ani není nalezena nově vytvořená hra, jedná se o chybu a uživateli je vrácen stav "ERROR."

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **game ID** - identifikace požadované hry
- **level** - požadovaná obtížnost hry (1 - lehká, 2 - normální, 3 - obtížná)
- **language** - požadovaný jazyk hry

Návratové hodnoty:

- **status = WAITING** - protihráči dosud nebyli nalezeni a uživatel je stále v poolu
- **status = MATCH_CREATED** - protihráči byli nalezeni, a byla vytvořena hra
- **status = ERROR** - došlo k chybě (uživatel není v poolu a hra nebyla vytvořena)
- **time = "unix timestamp"** - čas odpovědi (standardní UNIX timestamp)

Navazující zprávy:

- **pool-status** - vhodní protihráči nebyli nalezeni, a uživatel je tedy stále v poolu
- **game-accept** - vhodní protihráči byli nalezeni, a hra je tak vytvořena automaticky (uživatel byl přesunut z poolu)
- **pool-leave** - uživatel se rozhodl opustit pool (nechce už čekat na hru)

10.2.3. pool-leave

Pokud se uživatel rozhodne opustit pool (nechce už čekat na hru), měl by to pomocí této zprávy oznámit na server (aby nebyl nesprávně zařazován do nových her).

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **game ID** - identifikace hry z jejíhož poolu chce uživatel vystoupit
- **level** - požadovaná obtížnost hry (1 - lehká, 2 - normální, 3 - obtížná)
- **language** - požadovaný jazyk hry

Návratové hodnoty:

- **status = REMOVED** - uživatel byl vyjmut z poolu
- **status = ERROR** - došlo k chybě (uživatel není v poolu)
- **time = "unix timestamp"** - čas odpovědi (standardní UNIX timestamp)

10.2.4. game-accept

Po vytvoření musí všichni hráči hru přijmout (souhlasit že ji chtějí hrát) prostřednictvím této zprávy.

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **match ID** - identifikace hry (ze session)
- **accept = YES** - hra má být přijata
- **accept = NO** - hra má být odmítnuta

Návratové hodnoty:

- **status = ACCEPTED** - hráč hru přijal, přechází na "game-start"
- **status = DECLINED** - hráč hru zamítl, přechází na "pool-enter"
- **status = ERROR** - došlo k chybě (hra neexistuje, apod.)
- **time = "unix timestamp"** - čas odpovědi (standardní UNIX timestamp)

Navazující zprávy:

- **pool-enter** - hráč hru zamítl, a přechází tedy zpět do poolu
- **game-start** - hráč hru přijal, a čeká tedy na její zahájení

10.2.5. game-start

Pokud uživatel hru přijme, odchází čekat na zahájení hry (než hru přijmou i ostatní hráči). Pokud všichni hráči hru přijali, vrací se stav "ACCEPTED" a "start" obsahuje čas kdy hra začne (tak aby se všichni hráči mohli zkoordinovat). Pokud kterýkoliv hráč hru zamítne, vrací se stav "DECLINED"

a pokud ještě neodpověděli všichni hráči (a žádný zatím hru nezamítl), vrací se stav “WAITING.”

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **match ID** - identifikace hry (ze session)

Návratové hodnoty:

- **status = DECLINED** - některý hráč hru zamítl, přechází na “pool-status”
- **status = ACCEPTED** - všichni hráči hru přijali, čeká se na zahájení
- **status = WAITING** - žádný hráč hru neodmítl, ale některý zatím neodpověděl
- **status = ERROR** - došlo k chybě (hra neexistuje, apod.)
- **time = “unix timestamp”** - čas odpovědi (standardní UNIX timestamp)
- **start = “unix timestamp”** - kdy má být hra započata (jen pokud ji všichni přijmou)
- **words** - definice hádané věty (pouze pokud již hra byla odstartována a nebyla ukončena)

Navazující zprávy:

- **game-status** - jaký je aktuální stav hry (stav dalších hráčů, atd.)
- **game-guess** - hádání slova

Definice věty (**words**) je odesílána ve formátu kde jednotlivé části jsou oddělený znakem “|” (roura, pipe), na prvním místě je celkový počet slov ve větě a následují jednotlivé nápovědy ve tvaru “POZICE:SLOVO” (přičemž za nápovědu je považována i interpunkce). Přitom interpunkce je v definici obsažena vždy (není předmětem hádání).

Příkladem takto zakódované definice věty je například “5|1:Kdy|4:vyluxuješ|5:?” pro větu “Kdy už zatraceně vyluxuješ?”

10.2.6. game-status

V průběhu hry se uživatel může dotazovat na aktuální stav hry, tj. zejména zda hra probíhá (byla zahájena, skončila, ...), na současné výsledky ostatních hráčů, a podobně.

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **match ID** - ID zápasu (ze session)

Návratové hodnoty:

- **status = NOT_STARTED** - hra dosud nebyla zahájena
- **status = STARTED** - hra byla zahájena a dosud probíhá
- **status = FINISHED** - hra již skončila (použijte “game-results” pro získání výsledků)
- **status = ERROR** - došlo k chybě (hra neexistuje, apod.)
- **time = “unix timestamp”** - čas odpovědi (standardní UNIX timestamp)
- **results** - definice aktuálních výsledků pro jednotlivé protihráče (definice formátu dále)

Navazující zprávy:

- **game-status** - jaký je aktuální stav hry (stav dalších hráčů, atd.)
- **game-guess** - hádání slova

- **game-results** - jaké jsou celkové výsledky hry (po jejím skončení)

Formát položky “results” je obdobný jako v případě definice zadání věty, tj. na začátku je definice počtu hráčů, jednotliví hráči jsou odděleni dvěma znaky “||” a položky pro každého hráče jsou odděleny jedním znakem “|”. Jednotlivé položky vždy obsahují pozici hádaného slova v rámci věty, počet dosud využitých pokusů, a zda bylo hádání úspěšné či nikoliv (YES resp. NO). Položka “results” obsahuje informace i o aktuálním hráči, přičemž tento hráč je vždy uveden jako první.

Příkladem takto zakódovaného stavu může být řetězec “2||2:1:YES|4:3:NO||2:3:NO” udávající stav hry o dvou hráčích, kdy první hráč hádal dvě slova - slovo na druhé pozici uhádl na první pokus, a u slova na třetí pozici vyčerpal všechny tři pokusy - zatímco druhý hráč uhodl druhé slovo na třetí pokus.

10.2.7. game-guess

Tato zpráva se využívá pro odeslání informace o pokusu o uhádnutí slova v aktuální hře.

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **match ID** - ID zápasu (ze session)
- **sentence ID** - ID věty (ze session)
- **position** - pozice hádaného slova ve hře
- **word** - slovo hádané na pozici “position”

Návratové hodnoty:

- **result = CORRECT** - pokus o uhádnutí byl úspěšný
- **result = TOO_MANY_ATTEMPTS** - neúspěšný pokus (příliš mnoho pokusů)
- **result = INCORRECT_GUESS** - neúspěšný pokus (nesprávné slovo)
- **result = ERROR** - došlo k chybě (hra neexistuje, špatná pozice slova, apod.)
- **attempt = 1** - pořadí pokusu (1 až 3)
- **time = “unix timestamp”** - čas odpovědi (standardní UNIX timestamp)
- **correct = “správná odpověď”** - správná odpověď (při správné odpovědi, po 3 pokusu)
- **done = “(true|false)”** - je hra dokončena nebo ne? pokud ano, tak na game-result

Navazující zprávy:

- **game-status** - jaký je aktuální stav hry (stav dalších hráčů, atd.)
- **game-guess** - hádání slova
- **game-results** - jaké jsou celkové výsledky hry (po jejím skončení)

10.2.8. game-results

Tato zpráva se využívá pro získání informací o výsledcích skončené hry, tj. zejména správném znění věty a bodovém ohodnocení jednotlivých hráčů.

Vstupní parametry:

- **user ID** - ID uživatele (ze session)
- **match ID** - ID zápasu (ze session)

Návratové hodnoty:

- **result = OK** - ok, výsledky jsou vraceny
- **result = NOT_FINISHED** - hra ještě neskončila
- **result = ERROR** - došlo k chybě (hra neexistuje, apod.)
- **time = "unix timestamp"** - čas odpovědi (standardní UNIX timestamp)
- **sentence** - správné znění věty (slova oddělena znakem "|")
- **results** - výsledky jednotlivých hráčů, zakódované stejně jako v případě "game-status"
- **points** - informace o počtu bodů získaných jednotlivými hráči (kódování viz. níže)

Navazující zprávy:

- **pool-enter** - uživatel chce znovu vstoupit do poolu

Informace o počtu bodů jsou zakódovány takto - první číslo obsahuje informaci o počtu bodů získaných současným hráčem, a následují počty bodů dalších hráčů oddělené znakem "|". Příkladem takto zakódovaných informací může být "10|30|20" udávající informace o třech hráčích, přičemž aktuální hráč získal 10 bodů, druhý hráč získal 30 bodů a třetí získal 20 bodů.

10.3. Konfigurace PHP

Konfiguraci v rámci PHP vrstvy lze rozdělit na dvě části - konfiguraci vlastního PHP modulu tak aby se aplikace mohla správně spustit (zejména definice adresářů ze kterých se mají načítat knihovny), a nastavení vlastní aplikace (přístupy do DB, apod.).

10.3.1. Nastavení PHP modulu

Pro správnou funkci aplikace je třeba následující nastavení PHP modulu, které je možné provést v souborech php.ini nebo .htaccess.

PHP direktiva	hodnota
include_path	musí směřovat na adresáře "inc" a "libs" v kořenovém adresáři webu
register_globals	off
magic_quotes	off
short_open_tag	off

10.3.2. Nastavení PHP aplikace

Nastavení na úrovni PHP aplikace jsou uložena v souboru "inc/config.php" a jsou následující:

- spojení na databázi
  - DB_HOSTNAME - hostname databázového serveru
  - DB_PORT - port pro připojení k databázovému serveru
  - DB_DATABASE - název databáze
  - DB_SCHEMA - schéma v databázi
  - DB_USERNAME - uživatelské jméno pro připojení k databázi
  - DB_PASSWORD - heslo pro připojení k databázi
- logování chyb
  - CUST_ERROR_HANDLER - použít vlastní error handler (pokud "false" ztrácí následující nastavení význam)

        - LOGS_DIR - adresář do kterého mají být ukládány logy chyb
        - ERR_LOG_FILE - logovat chyby do souboru
        - ERR_LOG_FILE_LEVEL - chyby jakých úrovní logovat (viz. error_reporting)
        - ERR_LOG_FILE_DIR - adresář pro ukládání chyb PHP
        - ERR_LOG_FILE_PATH - soubor do kterého jsou ukládány chyby z PHP
        - ERR_LOG_MAIL - zasílat chyby e-mailem
        - ERR_LOG_MAIL_LEVEL - chyby jakých úrovní posílat e-mailem
        - ERR_LOG_MAIL_ADDR - na jakou adresu zasílat chyby
        - ERR_LOG_MAIL_SUBJ - předmět e-mailu s chybou
        - ERR_LOG_CONTEXT - ukládat / zasílat i PHP kontext (proměnné apod.)
- nastavení poolu
    - POOL_TIMEOUT - doba ve vteřinách po kterou je záznam v poolu aktivní
- nastavení úrovní obtížnosti
    - LEVEL_EASY_SENTENCE_MIN - min. počet slov pro lehkou obtížnost (3)
    - LEVEL_EASY_SENTENCE_MAX - max. počet slov pro lehkou obtížnost (5)
    - LEVEL_EASY_HINT_RATIO - procento nápověd pro lehkou obtížnost (50%)
    - LEVEL_MEDIUM_SENTENCE_MIN - min. počet slov pro střední obtížnost (4)
    - LEVEL_MEDIUM_SENTENCE_MAX - max. počet slov pro střední obtížnost (7)
    - LEVEL_MEDIUM_HINT_RATIO - procento nápověd pro střední obtížnost (30%)
    - LEVEL_HARD_SENTENCE_MIN - min. počet slov pro těžkou obtížnost (6)
    - LEVEL_HARD__SENTENCE_MAX - max. počet slov pro těžkou obtížnost (10)
    - LEVEL_HARD_HINT_RATIO - procento nápověd pro těžkou obtížnost (20%)
- bodové ohodnocení
    - CORRECT_FIRST_ATTEMPT - počet bodů za uhodnutí na první pokus (40)
    - CORRECT_SECOND_ATTEMPT - počet bodů za uhodnutí na druhý pokus (20)
    - CORRECT_THIRD_ATTEMPT - počet bodů za uhodnutí na první pokus (10)
    - INCORRECT_ATTEMPT - počet bodů za neuhodnutí (-10)
- výchozí jazyk
    - DEFAULT_LANGUAGE - cs

Pro jazykové verze jsou důležité soubory “inc/strings.XX.php” obsahující překlady všech textů na stránkách.

10.4. Flashová vrstva

Dodané zdrojové soubory vyžadují Flash CS3, finální SWF soubor vyžaduje Flash Player 6. Zdrojový soubor obsahuje 7 scén, přičemž každá implementuje jednu část aplikace.

10.4.1. Engine

Tato scéna obsahuje kód pro komunikaci se serverovým API a další pomocné funkce.

Engine komunikuje se serverovým API prostřednictvím HTTP POST požadavku (pomocí LoadVars objektu). K tomu slouží globální funkce "api(out, in, func)". Parametr "out" je objekt s proměnnými, které se mají předat funkci na serveru. Parametr "in" je objekt, do kterého se uloží odpověď volané funkce. Parametr "func" je název funkce serverového API např. "game-status".

Serverové API je v prototypu hry implementováno v jazyce PHP. Engine lze nakonfigurovat, aby používal i jinou implementaci. Nastavení adresy, kde je serverové API dostupné a jakého je typu, je v souboru shannon.xml.

10.4.2. Setup

Tato scéna obsahuje inicializační kód. Ten zpracuje konfigurační soubor.

10.4.3. Select Level

Tato scéna představuje obrazovku výběru obtížnosti hry.

10.4.4. Pool

Tato scéna představuje obrazovku, kdy hráč čeká na protihráče.

10.4.5. Game

Tato scéna představuje obrazovku samotné hry, kde hráč hádá slova. Zobrazení jednotlivých slov a jejich rozmístění zajišťuje instance "wordsclip" v události onEnterFrame.

10.4.6. Results

Tato scéna představuje obrazovku s výsledky hry.

10.4.7. Error

Tato scéna představuje obrazovku s chybovým hlášením.

10.5. Pracnost, finanční a časové odhady

Následující tabulka obsahuje stručné shrnutí odhadů časové a finanční náročnosti úvodní části projektu, popsané v předchozích odstavcích.

Název činnosti	Trvání	Zodpovědnost	Pracnost (h)	Cena (Kč)
Úvodní analýza, specifikace		Tomáš Vondra	10	2000
PHP a SQL část prototypu		Tomáš Vondra	40	8000
Tvorba Flashové části		David Šťastný	40	10000
Import data z csts formátu		Tomáš Vondra	10	2000

11. Otevřené otázky