



**MATEMATICKO-FYZIKÁLNÍ
FAKULTA**
Univerzita Karlova

ROČNÍKOVÁ PRÁCE

Jméno Příjmení

Závěrečná zpráva z ročníkového projektu

Název katedry nebo ústavu

Vedoucí bakalářské práce: Vedoucí práce

Studijní program: studijní program

Studijní obor: studijní obor

Praha 2024

Obsah

Úvod	3
1 Sada nástrojů	4
1.1 MM-cat	4
1.2 MM-infer	4
1.3 MM-evocat	6
1.4 MM-quecat	6
1.5 Nápady k této sekci, dočasná kapitola	6
2 Cílové skupiny uživatelů	7
2.1 Definice cílových skupin	7
2.2 Popis cílových skupin	8
2.2.1 Skupina mladých lidí	8
2.2.2 Skupina začínajících ve věku 40-50 let	8
2.2.3 Skupina expertních uživatelů	8
3 HTA (Hierarchical Task Analysis)	10
3.1 Vytváření diagramů	10
4 Storyboardy	12
4.1 Ukázka vytváření storyboardu	12
5 Low-fidelity prototyp	15
5.1 Paper prototyping	15
5.2 Low-fidelity návrh	15
5.3 Uživatelské testování	15
Závěr	17
Seznam použité literatury	18
Seznam obrázků	19
Seznam tabulek	20
Seznam použitých zkratk	21
A Přílohy	22
A.1 První příloha	22

Úvod

V rámci GAČR projektu Unified Management of Multi-Model Data (č. 20-22276S) byla navržena sada nástrojů pro modelování a správu multi-modelových dat, např. MM-cat a MM-evocat. Nástroje vznikaly postupně a za podpory různých lidí. Díky tomu nemají jednotné rozhraní pro práci. Navíc jsou současná rozhraní často nepřehledná a uživatelsky nepřívětivá. Proto se v rámci ročníkového projektu zabýváme návrhem sjednoceného rozhraní. Důraz je kladen na návrh vhodného uživatelského rozhraní a jeho udržitelnost.

Práce je soustředěna hlavně na nástroj MM-evocat. A to tak, aby bylo možné rozšířit aplikaci o další nástroje. Výsledkem práce je návrh low-fidelity prototypu MM-evocat a jeho otestování na vybraném vzorku reálných uživatelů.

TODO: nahradit mezery za no-break space všude v dokumentu... pomocí ~

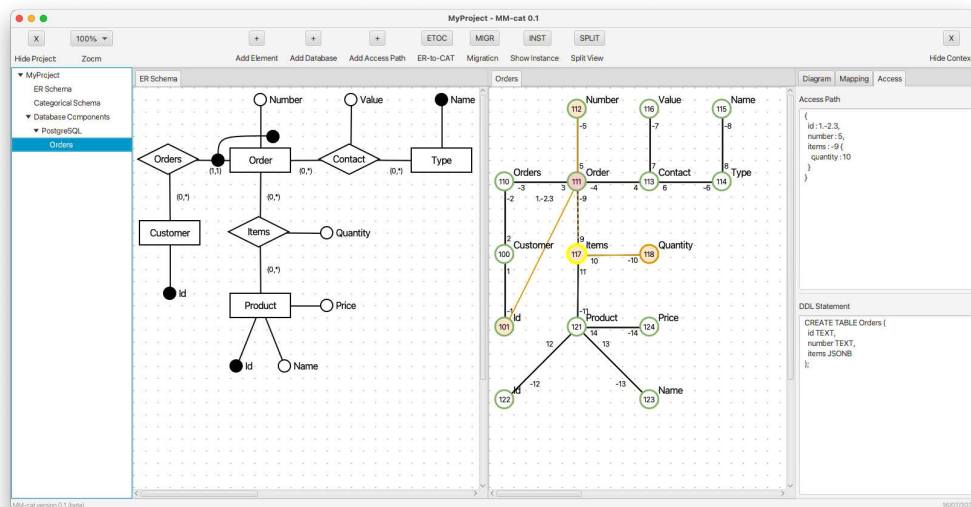
1. Sada nástrojů

1.1 MM-cat

MM-cat framework je navržený tak, aby řešil složitosti spojené s návrhem a správou multi-modelových databází. Jeho hlavním úkolem je modelování multi-modelových schém a jejich mapování na příslušné DBMS (Database Management System). Slouží i jako základ pro rozšíření o složitější úkoly.

Typickým scénářem použití je vytvoření ER (entity-relationship) schématu uživatelem. Takové schéma je pak automaticky převedeno do jednotné kategorické reprezentace, která umožňuje namapování na kombinaci DBMS. Díky specifikaci schématu je vytvořen skript s příkazy CREATE, které se aplikují na přiřazené DBMS. Uživatel může následně dále upravovat ER diagram, nebo provádět SELECT dotazy.

Na ukázce uživatelského rozhraní (Obrázek 1.1) je výsledek typického scénáře rozebraného v předchozím odstavci. Na levé straně je uživatelem vytvořený ER diagram, na straně pravé pak jeho kategorická reprezentace. Na pravé straně ještě stojí za povšimnutí panel s přístupovými cestami (značené oranžově) a CREATE příkaz.



Obrázek 1.1: Ukázka uživatelského rozhraní nástroje MM-cat.

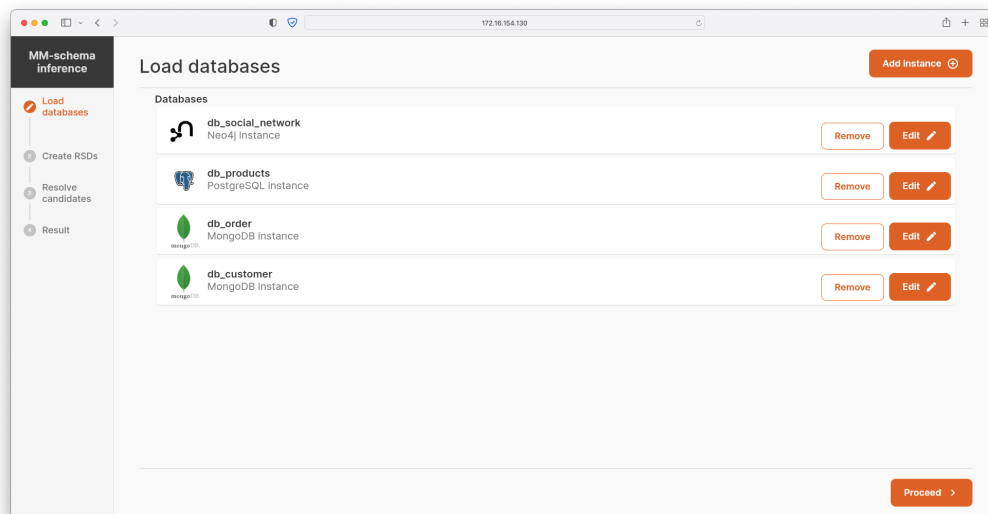
Další možnosti použití, i podrobnější popis nástroje, lze nalézt v článku [1].

1.2 MM-infer

Ne všechna data mají předem definované schéma. Nástroj MM-infer se snaží schéma zpětně zrekonstruovat z již uložených multi-modelových dat. Je schopen odhalit intra- a inter-modelové reference a překrývání modelů. Zároveň nástroj umí efektivně zpracovávat velké množství dat.

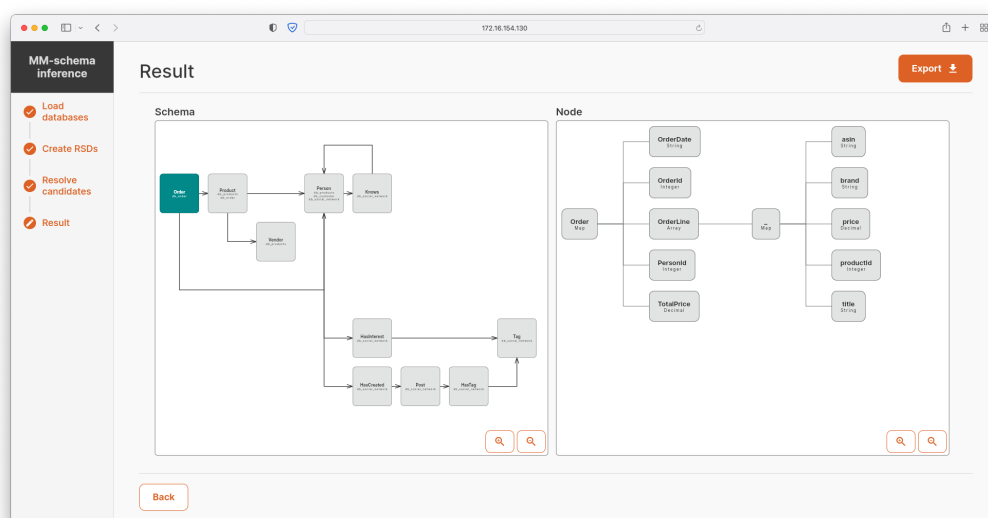
MM-infer podporuje tři druhy DBMS, vybrané tak, aby bylo pokryto co nejvíc funkcí takových systémů. PostgreSQL byl vybrán jako zástupce schema-full, relační DBMS. Neo4j reprezentuje schema-less a MongoDB obojí.

Rozhraní aplikace nás provede hned několika kroky, jejichž výsledkem je globální schéma pro vybranou množinu DBMS. Uživatel musí nejdříve takové DBMS vybrat (Obrázek 1.2). V levém panelu je osa ukazující již splněné a následující kroky, pro lepší orientaci v procesu.



Obrázek 1.2: Načtení databází v nástroji MM-infer.

Podle vybraných databází a dalších parametrů je vytvořen RSD (Record Schema Description), sjednocující reprezentace. Po automatickém načtení je pak předána ruka uživateli, který kontroluje navržené kandidáty. Výsledkem je vizualizace globálního schématu (Obrázek 1.3).



Obrázek 1.3: Vizualizace globálního schématu v nástroji MM-infer.

Aplikace příjemně provádí uživatele všemi potřebnými kroky. Nemá zbytečně

příliš různých grafických prvků, které by uživatele zahlcovaly. Stejně jako mm-cat využívá dvou oken pro diagramy vedle sebe.

Další informace o návrhu a možnostech použití aplikace lze nalézt v [2].

1.3 MM-evocat

MM-evocat je nástroj pro modelování a správu evoluce v multi-modelových datech. Řeší případy, kdy se struktura dat v čase mění tak, aby například vyhovovala novým uživatelským požadavkům. Takové změny jsou pak propagovány napříč definovanými modely a datovými instancemi.

Umožňuje vytvářet kategorický model, umí ho dekomponovat a umožňuje uživateli, aby si zvolil mapování na podporované DBMS. MM-cat podporuje stejné druhy systémů jako framework MM-infer (PostgreSQL, Neo4j, MongoDB).

Kategorický model, mapování a základy konceptuálního modelování popisuje článek [3].

MM-evocat má grafické webové rozhraní.

TODO: rozepsat rozhraní.

1.4 MM-quecat

Více v [4].

TODO: dopsat vše ke quecat.

1.5 Nápady k této sekci, dočasná kapitola

The Importance of the User Interface (v knize Galitz, W. O. The Essential Guide to User Interface Design. Wiley, 2002., tahle kniha je uvedena jako zdroj předmětu <https://digi.fel.cvut.cz/courses/nur>) -> proč vlastně děláme lepší návrh, je tam i user memory a obecně jak pracuje uživatel.

Dost podobné Designing Interfaces (co doporučil Jáchym)

Napsat motivaci pro sjednocení všech nástrojů.

2. Cílové skupiny uživatelů

Pro dobré nastavení projektu si definujeme cílové skupiny uživatelů, kteří budou naši aplikaci používat. Pokud už k předchozím projektům, které se snažíme sjednotit, cílové skupiny existují, je dobré se s nimi alespoň seznámit.

Jakmile nastavíme potenciální cílové skupiny uživatelů, zvolíme si konkrétní persony. Pod **personou** si můžeme představit detailní popis fiktivní osoby, která reprezentuje cílovou skupinu. Persony nám pomohou lépe rozebrat ekonomický status a vlastnosti lidí ve skupinách. Nezapomínáme ani na jejich motivace a zájmy. Nesnažíme se vystihnout všechny ze skupiny, ale vybíráme si hlavně stereotypní vlastnosti a chování. Stejně jako kladné rysy a zájmy chceme vystihnout i co dotyčný nemá rád, případně vůči čemu je úplně odmítavý. O tom co jsou persony a proč se nám vyplatí je definovat se krátce zmiňuje Jenifer Tidwell v knize Designing Interfaces [5].

Pro práci si definujeme jenom persony, které reprezentují uživatele na které cílíme. Lze ale i definovat vylučující persony. Tedy takové, na které cílit nechceme.

Vytváření cílových uživatelů je důležitým nástrojem, který nás bude provádět v dalších krocích projektu. Díky konkrétním představám nás budou lépe napadat konkrétní řešení a realizace. Snažíme se uzpůsobit návrh personám a vyvolat v nich kladné emoce. Díky tomu získáme konkrétně zaměřený projekt, který nebude tolik odtržený od reálných uživatelů.

2.1 Definice cílových skupin

Protože k předchozím projektům, které se snažíme sjednotit, persony a skupiny neexistují, definovali jsme skupiny nové.

Nejdřív jsme potenciální uživatele rozdělili na **začínající a expertní**. Jedni očekávají jednoduché koncepty, protože neznají ty složitější. Seznamují se s doménou poprvé a proto bývají mnohem rychleji frustrování, či odrazení. Nemají zkušenost ani nadhled, kterým by se učili aplikaci používat rychleji. Tak by se dala v krátkosti vystihnout skupina začínajících. Druzí už mají nějakou zkušenost v oboru, jsou zvyklí na dotazovací jazyky. Nerozumí přímo teorii co stojí za aplikací, ale spíše se opírají o již zmíněnou zkušenost v oboru.

Bylo těžké najít jednotné zástupce skupiny začínajících. I proto jsme se rozhodli skupinu rozdělit na podskupinu mladých lidí, kteří mají více sklony k hravosti a experimentování při objevování nových konceptů a na podskupinu lidí ve věku 40 až 50 let, kteří se chtějí rekvalifikovat pro práci s daty, ale zatím nemají vhodné zkušenosti.

Nakonec jsme si ještě pohrávali s myšlenkou, že by aplikaci využíval stroj. Jednotlivé úkony by byly prováděny pomocí skriptů. Ačkoli se taková skupina nedá zařadit do skupin potenciálních uživatelů, je potřeba i s takovým využitím počítat. Nicméně ji dál nebudeme rozebírat.

2.2 Popis cílových skupin

S charakterizací skupin nám pomůže představa konkrétních person. Do skupiny mladých začínajících můžeme zařadit bakalářského studenta informatického oboru. Pod starším začínajícím uživatelem si představíme padesátiletou pracovníci České pošty, která se chce requalifikovat pro práci s daty. Člověk co vystudoval MFF UK a dále pracuje v oboru s daty, třeba učitel Datového inženýrství, bude reprezentovat zástupce expertní skupiny uživatelů.

Za pomoci person rozebereme jednotlivé skupiny potenciálních uživatelů. Každou skupinu krátce charakterizujeme, následně se snažíme vystihnout její chování. Nakonec se zamyslíme nad úkoly, které budou uživatelé v aplikaci dělat.

2.2.1 Skupina mladých lidí

Je skupinou začínajících laiků, kteří se s aplikací seznamují úplně poprvé. Nemají příliš zkušeností v oboru, neznají dotazovací jazyk. Budou pracovat hlavně s grafickým prostředím, které by jim mělo práci usnadnit a motivovat je.

Chovají se spontánně, rozhodují se impulzivně a experimentují. Z toho co dělají vyzařuje hravost. Tímto přístupem velmi rychle prozkoumají různá zákoutí aplikace a dobře otestují její funkčnost. Učí se velmi rychle, ale stejně tak je dokáže rychle odradit i maličkost. Lépe si představují věci vizuálně a ocení odezvu nebo odměnu za úkol, který provedou. Odměna pro ně může být rychlá zpětná vazba, třeba odezva na úkony v grafickém prostředí. Pro pohyb v prostředí využívají hlavně počítačovou myš. Ocení barevnost, různorodost a zajímavé podání.

V aplikaci budou využívat možnosti dotazování nad daty, jednoduché modelování schematických kategorií, bez přidávání mapování a dalších složitějších operací. Nebude se jich týkat převod konceptuálního znázornění dat do multimodelové databáze a funkce nástroje MM-infer.

2.2.2 Skupina začínajících ve věku 40-50 let

Je druhá skupina začínajících uživatelů. Je to skupina, co se typicky chce requalifikovat do práce s daty, ale mají zatím jen nevhodné zkušenosti. Jsou seznámeni se základní prací na počítači, umí používat webový prohlížeč, tabulkový a textový procesor. Aplikace by pro ně měla být ideální na přeučení. Nemusí se učit komplexní technologie a dál budou dělat úkony přes grafické rozhraní.

Narozdíl od mladých začínajících mají metodický přístup chování na webu, přistupují k věci konzervativně. Upřednostňují umírněné podání grafiky. Jsou pro ně důležité popsané konkrétní kroky a argumenty. Nepracují spontánně, dělají zadanou práci. Je potřeba servírovat jim množství práce postupně v dávkách, aby nepřišlo informační zahlcení a odrazení od aplikace. Stejně jako mladí začínající jsou klikací typ, k práci využívají počítačovou myš.

Úkony prováděné v aplikaci jsou podobné skupině mladých začínajících lidí.

2.2.3 Skupina expertních uživatelů

Poslední skupina, která už zná koncepty ve světě databází. Zvyklá na textové dotazovací jazyky, databázové modely. Jsou schopní psát si vlastní skripty a jinak automatizovat a zefektivnit práci.

Nechovají se spontánně a k aplikaci přistupují hlavně metodicky. Dokážou se lépe orientovat v aplikaci, díky zkušenosti z oboru. Jsou spíše tolerantní, nemají přehnané nároky na aplikaci. Vědí, že je složité přecházet na jiný systém, narozdíl od začínajících uživatelů, které odradí i maličkost. Více si potrpí na funkce, které jim urychlí práci. Myšleno možnost používat skripty a klávesové zkratky. Ocení střízlivé podání a strohá data. Pracuje jak s grafickým rozhraním, tak i textovým.

Expertní uživatel využije více funkcí aplikace. Není omezen jen na tvorbu schematické kategorie. Může přidávat mapování a joby, převádět konceptuální znázornění dat do multimodelové databáze. Stejně tak bude využívat funkce nástroje MM-infer.

3. HTA (Hierarchical Task Analysis)

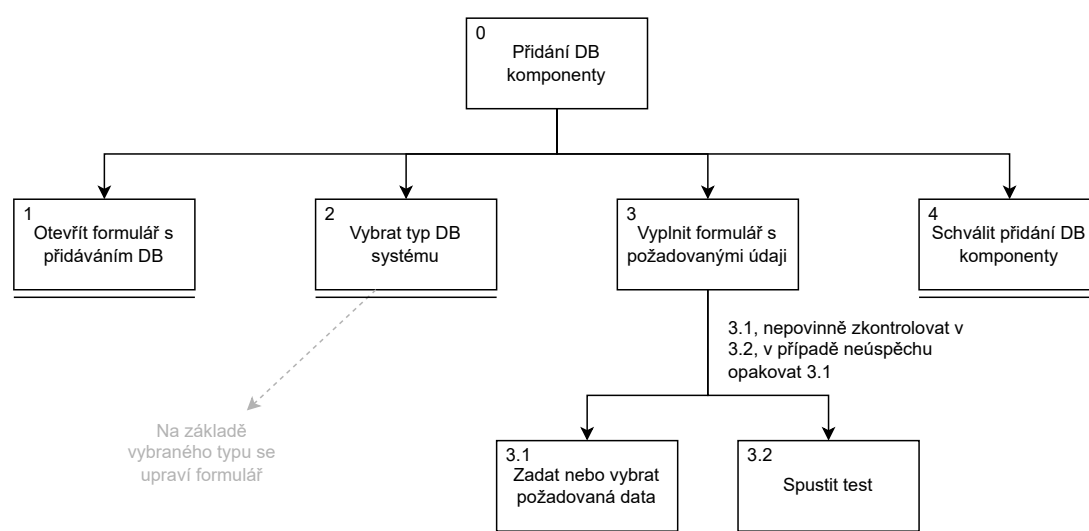
Na definování skupin uživatelů navážeme analýzou úkolů (Task Analysis) prováděných uživateli v aplikaci. Hlavním cílem analýzy je dokumentovat a strukturovat kroky nezbytné k dokončení úkolů. Díky definování různých úkolů, rozdělíme komplexní úlohy na menší, lépe představitelné části. Vše zaznamenejme do hierarchické struktury, odtud se vzalo slovo Hierarchical. Soustředíme se jen na pozitivní průchod.

Hierarchickou analýzu úkolů můžeme rozdělit na několik částí. Jako první určíme úkoly, které chtějí uživatelé v aplikaci dokončit. Každý hlavní úkol rozdělíme na podúkoly a akce, které definují vztah mezi podúkoly. Navíc každému hlavnímu úkolu přiřadíme výchozí situaci. Následně z jednotlivých komponent vytvoříme hierarchický diagram, v našem případě se stromovou strukturou. Jako poslední analyzujeme posloupnost akcí a snažíme se strukturu pochopit a vylepšit.

3.1 Vytváření diagramů

V charakterizaci cílových skupin (2.2) už jsme narazili na úkoly, které by uživatelé mohli dělat. Vybereme si z nich tři, které budeme analyzovat. Začneme úkolem, kdy uživatel přidá databázovou komponentu. Další hlavní úkol je vytvoření jednoduchého schématu a poslední jeho dekompozice. Vždy začínáme v situaci systému otevřeného na hlavní stránce.

Podívejme se na první jednoduchý diagram (Obrázek 3.1). Uživatel přidá nový databázový systém k již existujícím. Začíná ve spuštěném systému.



Obrázek 3.1: Diagram pro přidání databázové komponenty.

Kořen oznamuje jaký úkol bude uživatel provádět. V první hladině máme kroky 1 až 4, které nám v základních rysech ukazují, jak bude uživatel postupovat při plnění úkolu. Krok 3 dále rozdělíme na 3.1 Zadání dat a 3.2 Otestování. Na

hraně pak definujeme postup uživatele při plnění kroků 3.1 a 3.2. Když není hrana popsána, předpokládáme průběh kroků podle čísel v hladině vzestupně. Konkrétní pořadí ale není podmíněné. Číslování 3 a 3.x vyjadřuje vztah mezi rodičem a podúkoly. Podtržené stavy v HTA se dál nedělí.

Takto vytvořený diagram nám neříká nic o interakci uživatele s konkrétním systémem. Díky tomu ale můžeme kroky rychle měnit a upravovat. Lépe zhodnotíme, jestli uživatel nedělá příliš mnoho kroků pro dokončení úkolu, nebo naopak. Často se může stát, že jeden krok je ještě potřeba rozdělit.

4. Storyboardy

Na vytvoření uživatelských skupin a HTA diagramů navážeme tvořením Storyboardů. Storyboard je náčrt sekvence kroků, které uživatel v aplikaci provádí. Každý důležitý bod je zaznamenán do buňky. Ty pak skládáme lineárně za sebe, až nám vznikne příběh připomínající komiks. Takové rozdělení umožňuje soustředit se na každý krok zvlášť. Zároveň se jedná o další techniku, kde můžeme udělat rychlý náčrt a podle potřeby upravovat. Jeden storyboard nezabere víc než jednu stránku.

Cílem storyboardu je zjednodušeně zobrazit problém co se uživatel snaží v aplikaci řešit a zamyslet se nad jeho přístupem k danému problému. Jde nám hlavně o vizuální zobrazení konkrétních kroků, které jsme popisovali v HTA diagramech. Zároveň nám může být návrh nápomocný, abychom si vizualizovali, jak budou uživatelé používat navrhovanou aplikaci a jak bude aplikace v náznacích vypadat. Získané představy a vizualizace použijeme v detailnějším návrhu rozhraní a budeme se k nim snažit přiblížit.

V našem případě vezmeme reprezentanta každé skupiny uživatelů a představíme si jeho chování pro jednu vybranou situaci z námi vytvořeného HTA diagramu.

4.1 Ukázka vytváření storyboardu

Jako první začneme se zkušeným uživatelem. V aplikaci bude přidávat dvě různé databázové komponenty (PostgreSQL, MongoDB). Strukturu kroků zachováme podobnou prvnímu hta (3.1). Chceme zachytit uživatele, který se umí pohybovat v doméně, jenom nepoužíval náš nástroj.

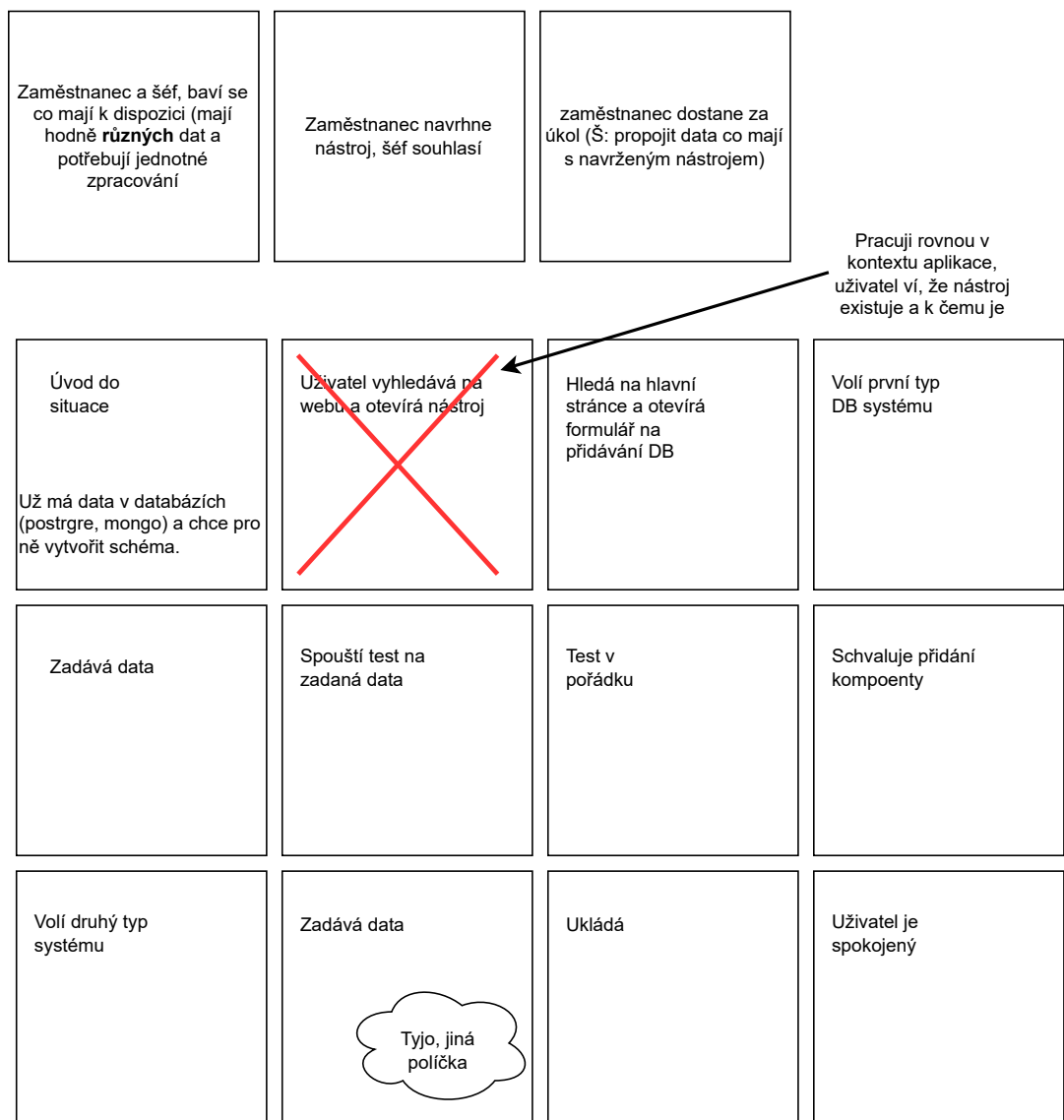
Na Obrázku 4.1 můžeme vidět rychlý návrh obsahu storyboardu. Kroky v buňkách nejdřív místo kreslení popisujeme. Získáme tím přehled všech hlavních kroků, které bude uživatel dělat a ujasníme si, kolik asi bude potřeba buněk.

V rámci návrhu jsme vytvořili několik iterací. Některé snímky jsme z návrhu vyřadili, protože nebyli potřebné (např. červeně škrtlý snímek). První tři kroky jsme k původnímu návrhu přidávali jako širší úvod do situace.

Dáváme si pozor, abychom se v této fázi návrhu nenechali strhnout detaily, které zatím nejsou potřeba. O prvotních fázích návrhu se zmiňuje i kniha [6]. Navrhuje jednoduchý způsob, jak se nenechat pohltnout detaily jako jsou typy písma, stíny, ikony atd. Podle autorů stačí vzít papír, tlustou fixu a tím si v podstatě znemožnit kreslení malých detailů. My jsme navíc omezení velikostí buněk, do kterých kreslíme.

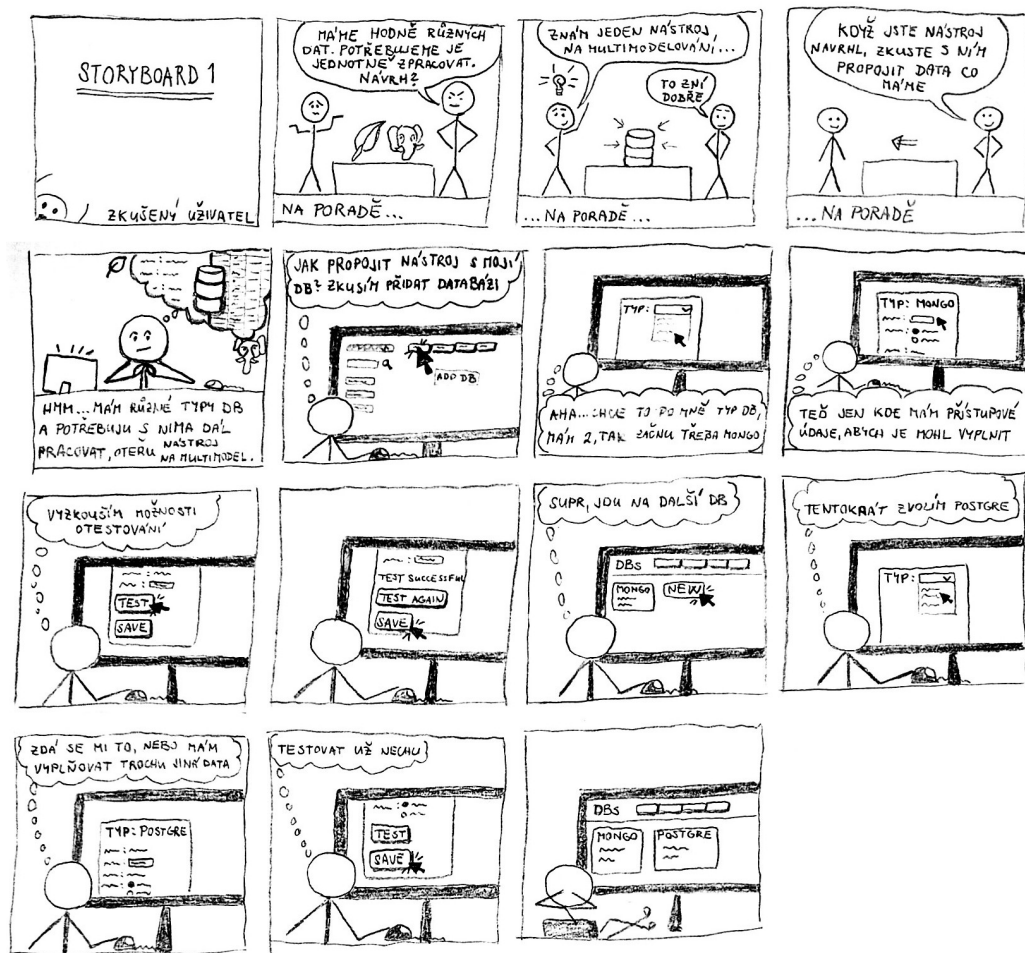
Při vyjadřování situace pamatujeme na to, že aplikaci momentálně ovládá zkušený uživatel. V aplikaci se zorientuje rychle a s úkolem nemá problém. Obrázek 4.2 je kompletní storyboard znázorněný pomocí kreseb místo textu. V návrhu je vidět, že uživatel nemá s používáním aplikace problém. Sám navrhl použití nástroje. Na několika snímcích je vidět i prvotní rozhraní aplikace, zatím opravdu jen v náznacích.

Oproti Obrázku 4.1 si díky vizualizaci v Obrázku 4.2 lépe představíme co bude uživatel dělat a jak bude v náznacích vypadat aplikace, ve které pracuje. Stejně tak se potom konkrétní představy lépe komunikují s lidmi se kterými storyboard



Obrázek 4.1: Rychlý návrh storyboardu pomocí textu v buňkách.

konzultujeme.



Obrázek 4.2: Výsledný storyboard, který znázorňuje uživatele přidávajícího databázové komponenty

5. Low-fidelity prototyp

UPDATE: Postupně přidávám další kapitoly. Není dokončené, zatím v procesu.

Do téhle kapitoly by se ještě mělo vejít porovnání různých nástrojů a proč jsme se tak rozhodli.

Po storyboardech, které nám ukázali možné sekvence akcí, vytvoříme prototyp aplikace, se kterým budou interagovat uživatelé. V rámci ročníkového projektu vytvoříme jen low-fidelity prototyp. Tedy se budeme zabývat vytvořením rané verze aplikace, která nebude věrnou kopií aplikace výsledné. V této fázi, chceme hlavně rychle vytvořit prototyp, na kterém můžeme vyhodnocovat nápady a zároveň ho můžeme otestovat na uživatelích spadajících do jednotlivých uživatelských skupin, které jsme si definovali.

5.1 Paper prototyping

Pro navrhování a testování uživatelského rozhraní, ve verzi low-fidelity, jsme si vybrali metodu prototypování na papír (paper prototyping). Celý proces od vytvoření prototypu až po uživatelské testování popisuje detailně kniha [7].

Začneme vytvořením papírových komponent aplikace (oken, menu, stránek, dialogových boxů, dat, pop-up zpráv). Po vytvoření prototypu provedeme usability otestování. V takovém testování provádí uživatel realistické úkoly na papírovém prototypu. Uživatelé jsou vybráni podle uživatelských skupin, které jsme si definovali na začátku. Testováním prováděným uživatelem myslíme jedno sezení s ním. Nejedná se o studii, kdy jsou prováděny série testů použitelnosti během několika dní.

Prototyp bude ovládaný osobou, která reprezentuje konání počítače, ale nevysvětluje, jak má rozhraní fungovat. Další zkušený člověk funguje jako zapisující pozorovatel. Pozoruje chování uživatele, zapisuje co uživatel dělá a co říká. Tímto způsobem provedeme velmi rychle iterativní testování na několika uživatelích. V průběhu můžeme aplikaci vylepšovat a všimnout si opakujících se vzorů. Každé sezení s uživatelem končí vyplněním dotazníku spokojenosti s aplikací.

5.2 Low-fidelity návrh

Jak funguje návrh... V nezávislosti na existující aplikaci vytváříme. Vytváříme jen pro počítač, ne pro telefon, pro tablet jen pro prohlížení. Zaměřili jsme se na to, jestli jsou komponenty dostupné, jestli nic nemáte uživatele, když má na něco kliknout...

Ukázat několik aplikací a porovnat, proč jsme se rozhodli tak a tak to udělat (jedna z nich <https://app.sqldbm.com/PostgreSQL/DatabaseExplorer/Draft/>). F-pattern, kam se dostanu po kolika kliknutích,

5.3 Uživatelské testování

Do téhle kapitoly dát obrázek z průběhu

Proč je dobré takový návrh vůbec testovat na uživatelích? (krom toho, že ho můžeme rychle měnit) str. 58 z [7] No nitpicky (hnedopíšský) feedback. Pokud se chceme vyhnout hnedopíšské zpětné vazbě, může nám pomoci low-fidelity návrh. Komponenty nejsou jasně dané, uživatel se spíš zaměří na koncepty a funkčnost. Je totiž očividné, že jsme ještě nespecifikovali vzhled. Zároveň to povzbuzuje uživatele, aby nebyl pasivní, ale sám kreativně přemýšlel nad koncepty. Dokonce není neobvyklé měnit vzhled s uživatelem přímo při testování. Tím, že nedokončený návrh povzbuzuje ke kreativitě, se zabývali výzkumníci v článku [8].

Techniky pro pozorování uživatelů. Chceme se připravit, pro maximalizaci užitečnosti dat z otestování. Kroky pro pozorování uživatelů, tak abychom na konci věděli, kde mají uživatelé problém aplikaci používat. Připravit si úkoly, které budou uživatelé dělat. Mělo by se jednat o reálné úkoly, které budou uživatelé v aplikaci nejběžněji dělat. Dál najít uživatele co sedí na persony. Dávat si pozor, aby neznali aplikaci nebo naše názory na ni. Pro testování vybrat místo, které je tiché a bez zbytečných vnějších vyrušování. Popsat uživateli o co se jedná, že jsou zapojeni do raných fází návrhu. Zdůraznit, že testujeme aplikaci, ne uživatele. Poprosit uživatele aby přemýšleli nahlas, aby říkali to co jim přijde na mysl, v průběhu plnění úkolů. Díky tomu prozkoumáme jejich očekávání od produktu, taky jejich úmysly a jejich strategie řešení problémů. Upozornit, že nebudeme uživateli pomáhat při plnění úkolů. Je to nejlepší způsob jak zjistit jak uživatelé reálně interagují s aplikací. Po testování zodpovědět zbývající otázky od uživatele. Lze diskutovat, nějaké zajímavé chování, které uživatel měl při testování. Zeptat se na celkový dojem z aplikace. Tohle je popsáno v knize [9] (kapitola Some Techniques for Observing Users)

V dotazníku (a obecně) jenom otevřené otázky (abychom nikoho nenaváděli).

Závěr

Seznam použité literatury

- [1] Pavel Koupil, Martin Svoboda, and Irena Holubová. MM-cat: A Tool for Modeling and Transformation of Multi-Model Data using Category Theory. In *Proc. of MODELS '21*, pages 635–639. IEEE, 2021.
- [2] Pavel Koupil, Sebastián Hricko, and Irena Holubová. MM-infer: A Tool for Inference of Multi-Model Schemas. In *Proc. of EDBT '22*, pages 2:566–2:569. OpenProceedings.org, 2022.
- [3] Pavel Koupil, Jáchym Bártík, and Irena Holubová. MM-evocat: A Tool for Modelling and Evolution Management of Multi-Model Data. In *Proc. of CIKM '22*, pages 4892–4896. ACM, 2022.
- [4] Pavel Koupil, Daniel Crha, and Irena Holubová. MM-quecat: A Tool for Unified Querying of Multi-Model Data. In *Proc. of EDBT '23*, pages 831–834. OpenProceedings.org, 2023.
- [5] J. Tidwell and an O'Reilly Media Company Safari. *Designing Interfaces, 2nd Edition*. O'Reilly Media, Incorporated, 2010.
- [6] A. Wathan and S. Schoger. *Refactoring UI*. Adam Wathan & Steve Schoger, 2019.
- [7] Carolyn Snyder. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. The Morgan Kaufmann series in interactive technologies. Elsevier Science, 2003.
- [8] Jutta Schumann, Thomas Strothotte, Andreas Raab, and Stefan Laser. Assessing the Effect of Non-photorealistic Rendered Images in CAD. In *Proceedings of CHI'96 Conference on Human Factors in Computing Systems (Vancouver, Canada, apr 1996)*, pages 35–42. ACM SIGCHI, 1996.
- [9] Brenda Laurel and S.J. Mountford. *The Art of Human-computer Interface Design*. Computer systems. Addison-Wesley, 1990.

Seznam obrázků

1.1	Ukázka uživatelského rozhraní nástroje MM-cat.	4
1.2	Načtení databází v nástroji MM-infer.	5
1.3	Vizualizace globálního schématu v nástroji MM-infer.	5
3.1	Diagram pro přidání databázové komponenty.	10
4.1	Rychlý návrh storyboardu pomocí textu v buňkách.	13
4.2	Výsledný storyboard, který znázorňuje uživatele přidávajícího da- tabázové komponenty	14

Seznam tabulek

Seznam použitých zkratek

A. Přílohy

A.1 První příloha