



NOME COMPLETO DO AUTOR

Licenciado em Engenharia Informática

SINCRONIZAÇÃO DE DOCUMENTOS NA EDIÇÃO COLABORATIVA EM TEMPO REAL

MESTRADO EM ENGENHARIA INFORMÁTICA

Universidade NOVA de Lisboa
Outubro, 2026

SINCRONIZAÇÃO DE DOCUMENTOS NA EDIÇÃO COLABORATIVA EM TEMPO REAL

NOME COMPLETO DO AUTOR

Licenciado em Engenharia Informática

Orientador: Nome do Orientador

Professor Auxiliar, Universidade NOVA de Lisboa

Coorientadora: Nome da Coorientadora

Professora Associada, Universidade NOVA de Lisboa

Júri

Presidente: Nome do Presidente

Professor Catedrático, Universidade NOVA de Lisboa

Arguente: Nome da Arguente

Professora Auxiliar, Universidade de Lisboa

Orientador: Nome do Orientador

Professor Auxiliar, Universidade NOVA de Lisboa

MESTRADO EM ENGENHARIA INFORMÁTICA

Universidade NOVA de Lisboa

Outubro, 2026

Sincronização de documentos na edição colaborativa em tempo real

Copyright © Nome Completo do Autor, Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa.

A Faculdade de Ciências e Tecnologia e a Universidade NOVA de Lisboa detêm o direito perpétuo e sem limites geográficos de arquivar e publicar esta dissertação em formato impresso ou digital, ou por qualquer outro existente ou que venha a ser criado, bem como de a divulgar através de repositórios científicos e de permitir a sua cópia e distribuição para fins educacionais ou de investigação, não-comerciais, desde que seja dado o devido crédito ao autor e ao editor.

AGRADECIMENTOS

Os agradecimentos são um texto pessoal e uma expressão livre do autor. Começam habitualmente pelos agradecimentos académicos (orientadores), seguem-se os institucionais (centro de investigação, departamento, faculdade, bolsas) e, por fim, os pessoais (amigos, família).

RESUMO

O resumo apresenta em poucos parágrafos o contexto do trabalho, o problema abordado, a abordagem seguida e os principais resultados obtidos. Não deve ultrapassar uma página.

O segundo parágrafo pode indicar o alcance dos resultados, as suas limitações e as perspectivas de trabalho futuro.

Palavras-chave: Primeira palavra-chave · Segunda palavra-chave · Terceira palavra-chave

ABSTRACT

The abstract presents in a few paragraphs the context of the work, the problem addressed, the approach taken and the main results. It should not exceed one page.

The second paragraph may state the scope of the results, their limitations and the perspectives for future work.

Keywords: First keyword · Second keyword · Third keyword

ÍNDICE

Resumo	iv
Abstract	v
Índice	vi
Índice de Figuras	vii
Índice de Tabelas	viii
Úvod	1
1 Společná editace dokumentů	2
1.1 Požadavky na systém	3
2 Přístupy k synchronizaci	4
2.1 Operační transformace	4
2.1.1 Transformační funkce	4
2.2 Bezkonfliktní replikované datové typy	4
3 Návrh a implementace	6
3.1 Ukládání historie	6
4 Vyhodnocení	7
4.1 Diskuse výsledků	8
Závěr	9
Literatura	10

ÍNDICE DE FIGURAS

1.1	Typický průběh společného psaní dokumentu	2
4.1	Počet úprav podle dne v týdnu a hodiny ve sledovaných relacích	8

ÍNDICE DE TABELAS

4.1	Průměrná latence a velikost uložených dat podle počtu uživatelů	7
-----	---	---

ÚVOD

Textové dokumenty dnes jen výjimečně vznikají v rukou jediného autora. Studenti píší seminární práce ve dvojicích, vývojáři společně udržují technickou dokumentaci a výzkumné týmy dokončují články na poslední chvíli z několika míst najednou. Aby takové psaní fungovalo, musí editor zajistit, že všichni účastníci vidí stejný text a že se žádná úprava neztratí.

Cílem této práce je porovnat dva hlavní přístupy k synchronizaci dokumentů při společné editaci v reálném čase, operační transformaci a bezkonfliktní replikované datové typy, a ověřit je na prototypu webového editoru. První kapitola vymezuje požadavky na takový systém, druhá popisuje oba přístupy, třetí návrh prototypu a čtvrtá výsledky měření.

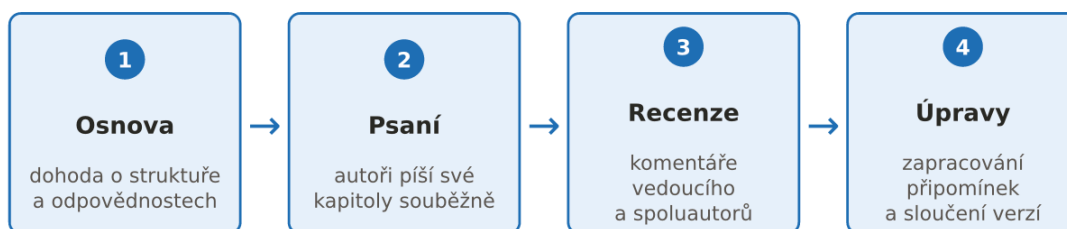
SPOLEČNÁ EDITACE DOKUMENTŮ

Společnou editací rozumíme situaci, kdy více uživatelů upravuje tentýž dokument a jejich změny se navzájem propisují bez ručního slučování. Podle toho, jak rychle se změny šíří, rozlišujeme tři základní režimy práce:

- **asynchronní editace**, při níž si autoři předávají celé verze dokumentu a slučují je až dodatečně,
- **editace se zámky**, kdy smí v jednu chvíli upravovat určitou část dokumentu jen jeden autor,
- **editace v reálném čase**, při níž se každá změna okamžitě promítne ostatním uživatelům.

Tato práce se zabývá výhradně posledním režimem. Ten je pro uživatele nejpohodlnější, ale klade největší nároky na návrh systému, protože změny vznikají souběžně na mnoha místech a přicházejí v různém pořadí.

Editace v reálném čase přitom nemění jen techniku, ale i průběh celé spolupráce. Typický postup společného psaní, jak jej popsali dotazovaní uživatelé, zachycuje [Figura 1.1](#).



kroky 2 až 4 se opakují, dokud recenze přináší podstatné připomínky

Figura 1.1: Typický průběh společného psaní dokumentu

1.1 Požadavky na systém

Z rozhovorů s uživateli existujících editorů a z odborné literatury jsme odvodili čtyři požadavky, které musí systém pro společnou editaci splnit:

1. konvergence, tedy po doručení všech změn mají všechny kopie dokumentu stejný obsah,
2. zachování záměru, tedy výsledek úpravy odpovídá tomu, co autor v okamžiku úpravy viděl,
3. odezva, tedy lokální úprava se projeví okamžitě bez čekání na server,
4. odolnost, tedy krátkodobý výpadek spojení nevede ke ztrátě rozepsaného textu.

První dva požadavky se týkají správnosti, zbylé dva použitelnosti. V praxi se ukazuje, že právě kombinace okamžité odezvy a zaručené konvergence je nejtěžší částí celého návrhu.

PŘÍSTUPY K SYNCHRONIZACI

Systémy pro společnou editaci v reálném čase si každou změnu nejprve provedou lokálně a teprve potom ji rozešlou ostatním účastníkům. Liší se především tím, jak řeší situaci, kdy dvě změny vzniknou souběžně a každý účastník je obdrží v jiném pořadí.

2.1 Operační transformace

Operační transformace vychází z práce Ellise a Gibbse [1], kteří jako první popsali, jak upravit souběžnou operaci tak, aby ji bylo možné provést i po jiné operaci, o které její autor nevěděl. Každá úprava se zapisuje jako operace vložení nebo smazání na určité pozici v textu.

Pokud například jeden uživatel vloží slovo na začátek odstavce a druhý současně smaže slovo na jeho konci, musí se pozice mazání posunout o délku vloženého slova. Tento posun počítá transformační funkce.

2.1.1 Transformační funkce

Transformační funkce dostane dvojici souběžných operací a vrátí jejich upravené verze. Aby všechny kopie dokumentu konvergovaly, musí funkce splňovat takzvané transformační vlastnosti, jejichž dodržení je pro obecné operace obtížné dokázat. Většina nasazených systémů proto spoléhá na centrální server, který určuje jediné pořadí operací.

2.2 Bezkonfliktní replikované datové typy

Druhý přístup představují bezkonfliktní replikované datové typy, souhrnně označované zkratkou CRDT [2]. Místo dodatečné úpravy operací volí takovou reprezentaci dokumentu, v níž souběžné operace komutují a jejich pořadí nemá na výsledek vliv.

Replikovaný datový typ je bezkonfliktní, pokud libovolné dvě repliky, které obdržely stejnou množinu aktualizací, mají stejný stav bez ohledu na pořadí jejich doručení.

Každý znak dokumentu dostane jednoznačný identifikátor, který zůstává stejný po celou dobu existence znaku. Pozice vložení se pak neurčuje číslem, ale odkazem na identifikátor sousedního znaku, takže ji souběžné úpravy jinde v textu nemohou posunout.

NÁVRH A IMPLEMENTACE

Prototyp editoru se skládá z webového klienta, synchronizačního serveru a databáze, do níž server průběžně ukládá stav dokumentů. Klient i server jsou napsány tak, aby bylo možné synchronizační vrstvu vyměnit a porovnat oba popsané přístupy na stejném uživatelském rozhraní.

Klient udržuje lokální repliku dokumentu a každou úpravu uživatele převede na operaci nad replikovaným datovým typem. Operace se odesílají serveru po dávkách, aby se při rychlém psaní nezahltla síť. Zjednodušené zpracování příchozí dávky na serveru ukazuje následující výpis:

```
def zpracuj_davku(dokument, davka):  
    for operace in davka.operace:  
        if operace.id in dokument.pouzite:  
            continue  
        dokument.aplikuj(operace)  
        dokument.pouzite.add(operace.id)  
    rozesli_ostatnim(dokument.id, davka)
```

Protože operace komutují, nemusí server dávky řadit ani transformovat. Stačí, když zajistí, že se žádná operace neprovede dvakrát a že každá dávka dorazí ke všem připojeným klientům.

3.1 Ukládání historie

Server neukládá jen výsledný text, ale celý log operací. Z logu lze obnovit libovolnou dřívější verzi dokumentu a znovu sestavit stav repliky klienta, který byl delší dobu odpojen. Tento přístup odpovídá architektuře založené na neměnném záznamu událostí, jak ji popisuje Kleppmann [3]. Aby log nerostl neomezeně, server jej v pravidelných intervalech zhušťuje do snímků.

VYHODNOCENÍ

Prototyp jsme porovnali s referenční implementací operační transformace na sadě záznamů skutečných editačních relací. Pro každou relaci jsme sledovali latenci, tedy dobu mezi úpravou u autora a jejím zobrazením u ostatních účastníků, a průměrnou latenci relace jsme spočítali podle (4.1):

$$\bar{t} = \frac{1}{n} \sum_{i=1}^n (t_i^{\text{doručení}} - t_i^{\text{úprava}}) \quad (4.1)$$

kde n je počet operací v relaci. Naměřené hodnoty pro různý počet souběžně pracujících uživatelů shrnuje [Tabela 4.1](#).

Tabela 4.1: Průměrná latence a velikost uložených dat podle počtu uživatelů

Uživatelé	Latence OT (ms)	Latence CRDT (ms)	Data (kB)
2	38	35	12
5	52	41	19
10	87	48	31
20	164	63	58

S rostoucím počtem uživatelů roste latence operační transformace výrazně rychleji, protože server musí každou operaci transformovat proti všem souběžným operacím. Cenou za nižší latenci CRDT je větší objem uložených dat, neboť každý znak nese svůj identifikátor. Rozložení úprav v čase, z něhož vychází zátěžový test, ukazuje [Figura 4.1](#).

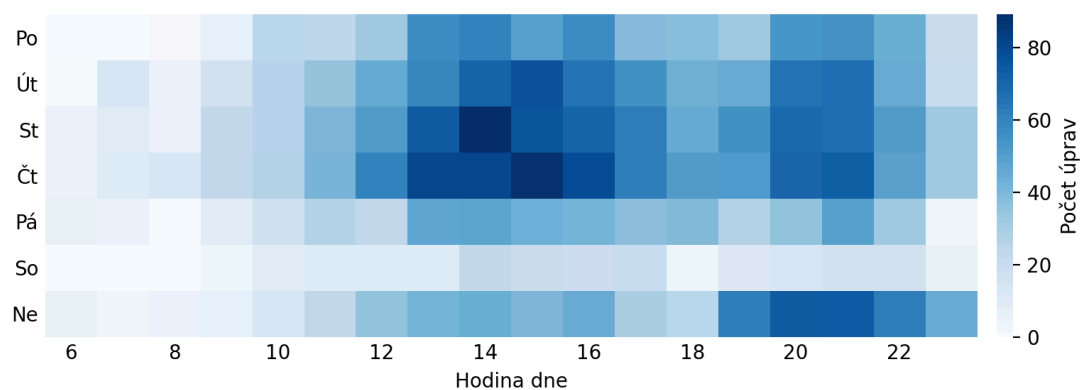


Figura 4.1: Počet úprav podle dne v týdnu a hodiny ve sledovaných relacích

4.1 Diskuse výsledků

Výsledky potvrzují, že pro menší týmy jsou oba přístupy rovnocenné a rozhoduje spíše jednoduchost implementace. Při větším počtu souběžných uživatelů má CRDT jasnou výhodu v latenci i v odolnosti vůči výpadkům spojení. Doplnující rozhovory s uživateli, vedené podle zásad kvalitativního výzkumu [4], ukázaly, že rozdíl v latenci pod sto milisekund uživatelé nevnímají, zato ztrátu rozepsaného textu po výpadku spojení považují za zásadní problém.

ZÁVĚR

Práce porovnála operační transformaci a bezkonfliktní replikované datové typy jako dva přístupy k synchronizaci dokumentů při společné editaci v reálném čase. Na prototypu webového editoru jsme ukázali, že CRDT umožňují jednodušší server bez centrálního řazení operací a při větším počtu uživatelů nabízejí nižší latenci.

Hlavní nevýhodou CRDT zůstává větší objem ukládaných dat. Dalším krokem proto bude ověřit, jak velikost dat ovlivňuje pravidelné zhušťování logu do snímků a jak se prototyp chová při dlouhodobém provozu s desítkami tisíc dokumentů.

LITERATURA

- [1] C. A. Ellis e S. J. Gibbs, «Concurrency control in groupware systems», em *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, New York: ACM, 1989, pp. 399–407. doi: [10.1145/67544.66963](https://doi.org/10.1145/67544.66963).
- [2] M. Shapiro, N. Preguiça, C. Baquero, e M. Zawirski, «Conflict-free replicated data types», em *Stabilization, Safety, and Security of Distributed Systems*, Berlin: Springer, 2011, pp. 386–400. doi: [10.1007/978-3-642-24550-3_29](https://doi.org/10.1007/978-3-642-24550-3_29).
- [3] M. Kleppmann, *Designing data-intensive applications*. Sebastopol: O'Reilly Media, 2017.
- [4] J. Hendl, *Kvalitativní výzkum: základní metody a aplikace*. Praha: Portál, 2005.

