universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Exploring ActivityPub Interoperability for Bluesky“

verfasst von / submitted by

Martin Sonnberger BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2026 / Vienna, 2026

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 935
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Medieninformatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Math. Dr. Peter Reichl, Privatdoz.

Acknowledgements

Thank you very much!!

Abstract

This LaTeX template provides example on how to format and display text, mathematical formulas, and insert tables or images. There is a lot more you can do with LaTeX, for more information check out https://en.wikibooks.org/wiki/LaTeX.

Kurzfassung

Das ist eine deutsche Kurzfassung meiner in Englisch verfassten Masterarbeit.

Contents

List of Tables

List of Figures

List of Algorithms

Listings

1. Introduction

2. Background and Related Work

2.1. AT Protocol

2.2. ActivityPub

2.3. Related Work

2.3.1. Bridgy Fed

2.3.2. Wafrn

3. System Architecture

3.1. Overview

3.1.1. Sidecar Pattern

Fedisky is designed as a *sidecar container* as defined in [1, Ch. 3] and runs alongside its *application container*, a Bluesky PDS. Sidecars are a common architectural pattern, where a secondary container provides auxiliary functionality to the primary application container, often without the application knowing. Sidecar containers share system resources and are scheduled to run in sync with the application container [1, p. 21]. This design allows Fedisky to operate independently while still closely integrating with the PDS.

An alternative design could have been a centralized bridging service that bridges *all* of Bluesky into the Fediverse, which is realized by Bridgy Fed (subsection 2.3.1). Compared to a centralized bridging service, a sidecar container running alongside a single Bluesky PDS instance, can share resources with the PDS, most importantly the server's hostname, which allows it to operate under the same domain and thus provide a single shared identity for users across Bluesky and Mastodon, e.g. `alice.fedisky.social` in Bluesky becomes `@alice@fedisky.social` in Mastodon. Additionally, the sidecar pattern allows for operator autonomy, where each PDS operator can choose to deploy Fedisky independently, or not. It also reduces the scope of the bridge, as it only needs to handle traffic for a single PDS instance, simplifying development and maintenance.

Another approach would be to fork and modify the PDS itself, thus integrating bridging functionality directly into the PDS. We considered this approach initially when designing Fedisky, but chose the sidecar pattern instead for several reasons. Forking the PDS would significantly increase the maintenance burden and development complexity, as one would need to keep up with upstream changes constantly to ensure compatibility. The sidecar avoids this by only consuming the PDS's public APIs, which are stable contracts based on AT Protocol Lexicons [2]. In addition, the sidecar pattern allows for better separation of concerns, as all bridging logic is contained within the sidecar, rather than the PDS being responsible for both AT Protocol and ActivityPub compliance and therefore tightly coupling two protocol implementations, making each harder to reason about and test in isolation. Furthermore, the sidecar gives more deployment flexibility, as operators can choose to start or stop the bridge independently of the PDS. Since communication between the sidecar and the PDS is done through HTTP APIs, the sidecar can be developed a different technology stack, and is automatically compatible with any PDS implementation that adheres to the AT Protocol specifications.

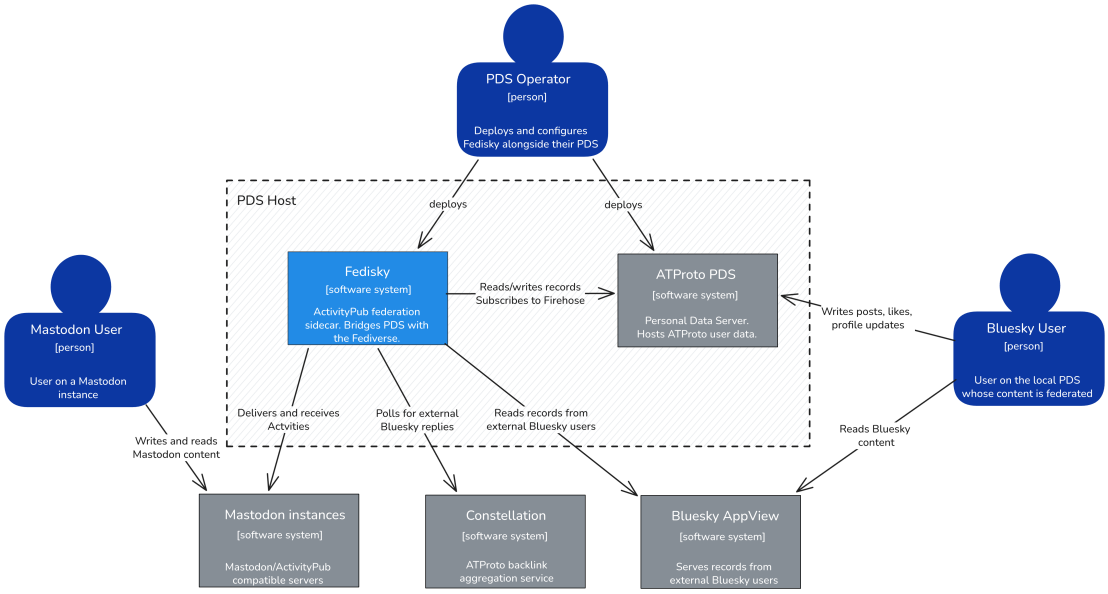

Figure 3.1.: System Context Diagram showing how the Fedisky sidecar interacts with other systems and users.

4. Results

5. Discussion

6. Conclusion

Bibliography

[1] Brendan Burns. *Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services, Using Kubernetes*. Second Edition. Sebastopol: O'Reilly, 2024. 200 pp. ISBN: 978-1-0981-5635-0.

[2] *HTTP Reference | Bluesky*. URL: `https://docs.bsky.app/docs/category/http-reference` (visited on 02/12/2026).

A. Appendix

here you can put further things you want to add like transcripts, questionnaires, raw data...