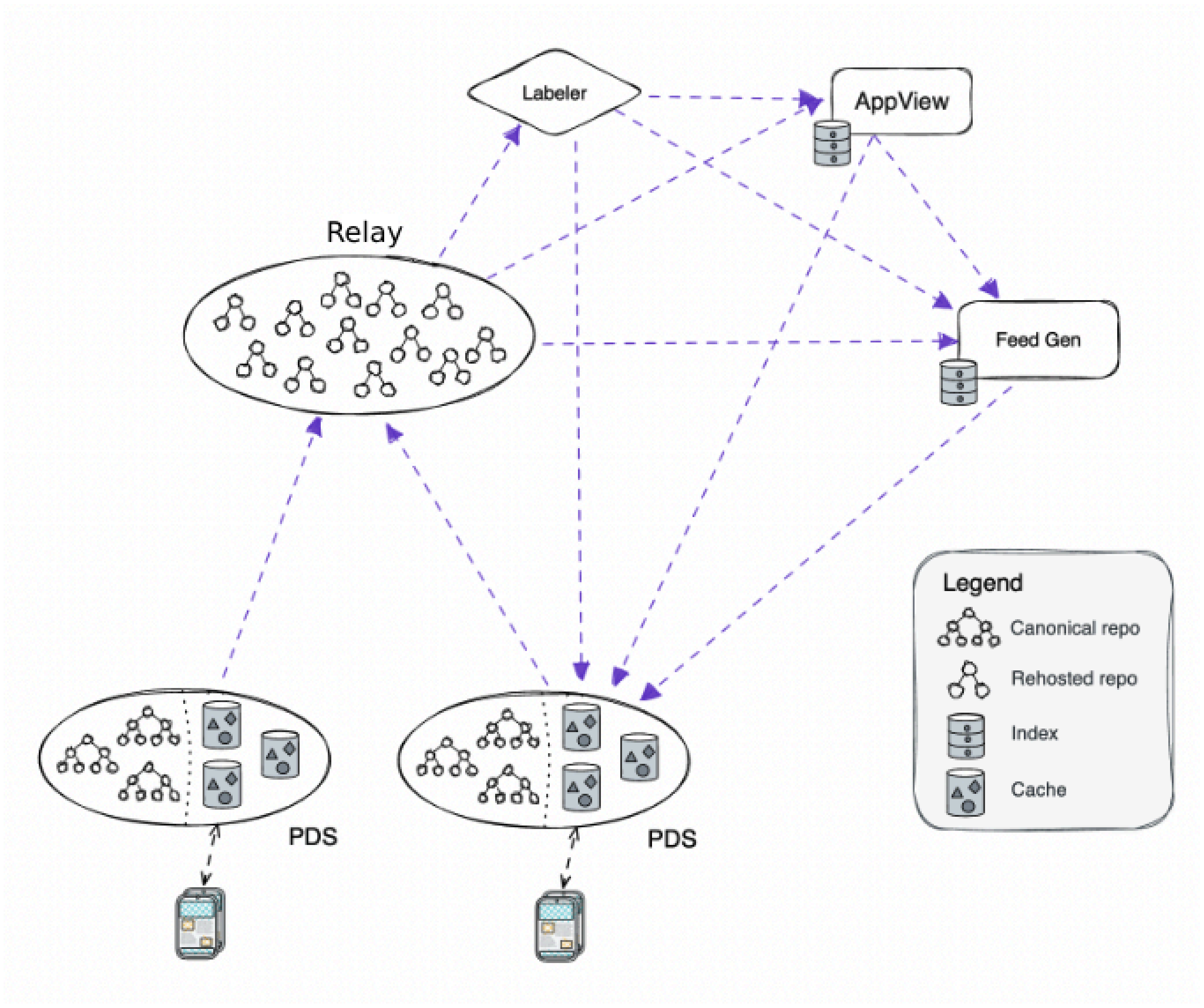
Federation Architecture

The AT Protocol is made up of a bunch of pieces that stack together. Federation means that anyone can run the parts that make up the AT Protocol themselves, such as their own server.

The three main services are personal data servers (PDS), Relays, and App Views. Developers can also run feed generators (custom feeds), and labelers are in active development.



Personal Data Server (PDS)

A PDS acts as the participant’s agent in the network. This is what hosts your data (like the posts you’ve created) in your repository. It also handles your account & login, manages your repo’s signing key, stores any of your private data (like which accounts you have muted), and handles the services you talk to for any request.

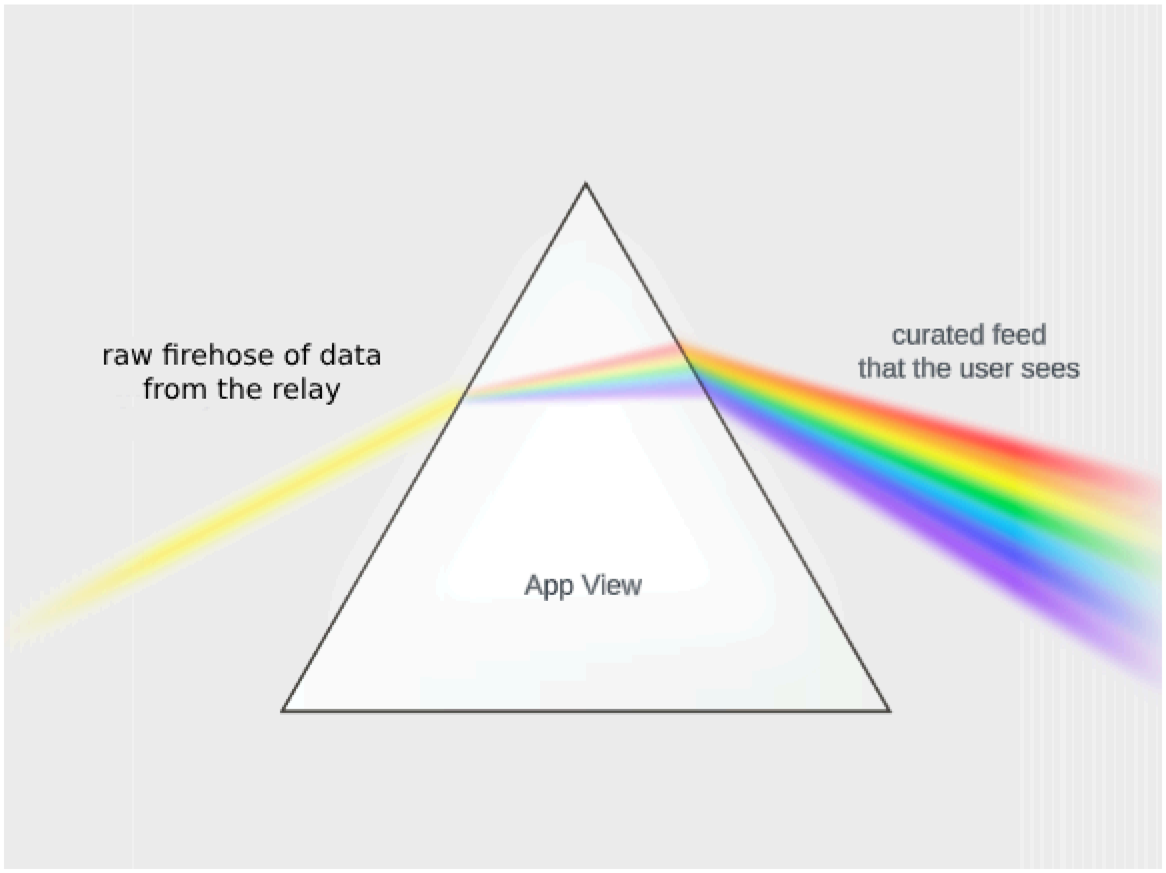
Relay

The Relay handles "big-world" networking. It crawls the network, gathering as much data as it can, and outputs it in one big stream for other services to use. It’s analogous to a firehose provider or a super-powered relay node.

Anyone can host a Relay, though it’s a fairly resource-demanding service. In all likelihood, there may be a few large full-network providers, and then a long tail of partial-network providers. Small bespoke Relays could also service tightly or well-defined slices of the network, like a specific new application or a small community.

App Views

An App View is the piece that actually assembles your feed and all the other data you see in the app, and is generally expected to be downstream from a Relay's firehose of data. This is a highly semantically-aware service that produces aggregations across the network and views over some subset of the network. This is analogous to a prism that takes in the Relay's raw firehose of data from the network, and outputs views that enable an app to show a curated feed to a user. For example, the Relay might crawl to grab data such as a certain post’s likes and reposts, and the app view will output the count of those metrics.


raw firehose of data from the relay
curated feed that the user sees
App View

There will also be an ecosystem of App Views for each lexicon, or “social mode,” deployed on the network. For example, Bluesky currently supports a micro-blogging mode: the `app.bsky` lexicon. Developers who create new lexicons would likely deploy a corresponding App View that understands their lexicon to service their users. Other lexicons could include video or long-form blogging, or different ways of organizing like groups and forums. By bootstrapping off of an existing Relay, data collation will already be taken care of for these new applications. They need only provide the indexing behaviors necessary for their application.

“Big World” Design

The AT Protocol is architected in a “big world with small world fallbacks” way, modeled after the open web itself. With the web, individual computers upload content to the network, and then all of that content is then broadcasted back to other computers. Similarly, with the AT Protocol, we’re sending messages to a much smaller number of big aggregators, which then broadcast that data to personal data servers across the network. Additionally, we solve the major problems that have surfaced from the web through self-certifying data, open schematic data and APIs, and account portability.

On a technical level, prioritizing big-world indexing over small world networking has multiple benefits.

- It significantly reduces the load on PDSs, making it easier to self-host — you could easily run your own server.
- It improves discoverability of content outside of your immediate neighbors — people want to use social media to see content from outside of their network.
- It improves the quality of experience for everyone in the network — fewer dropped messages or out-of-sync metrics.

Given all that, our proposed methodology here of networking through Relays instead of server-to-server isn't prescriptive. The protocol is actually explicitly designed to work both ways.

Self-Hosting

You can run your own PDS instance in the federated network! For source code and directions see the PDS distribution git repository here.

Edit this page