b
UNIVERSITY
OF BERN

Community-Driven Variability

Meeting on Feature-Oriented Software Development (FOSD)

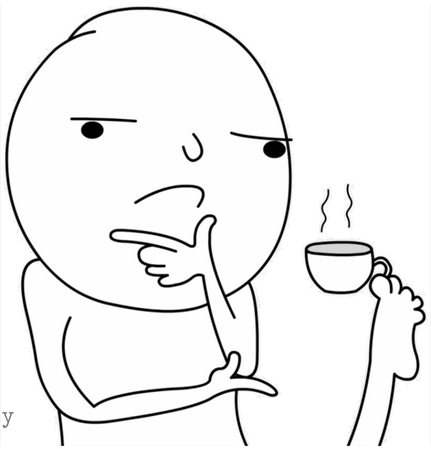
2025-03-25, Köthen DE

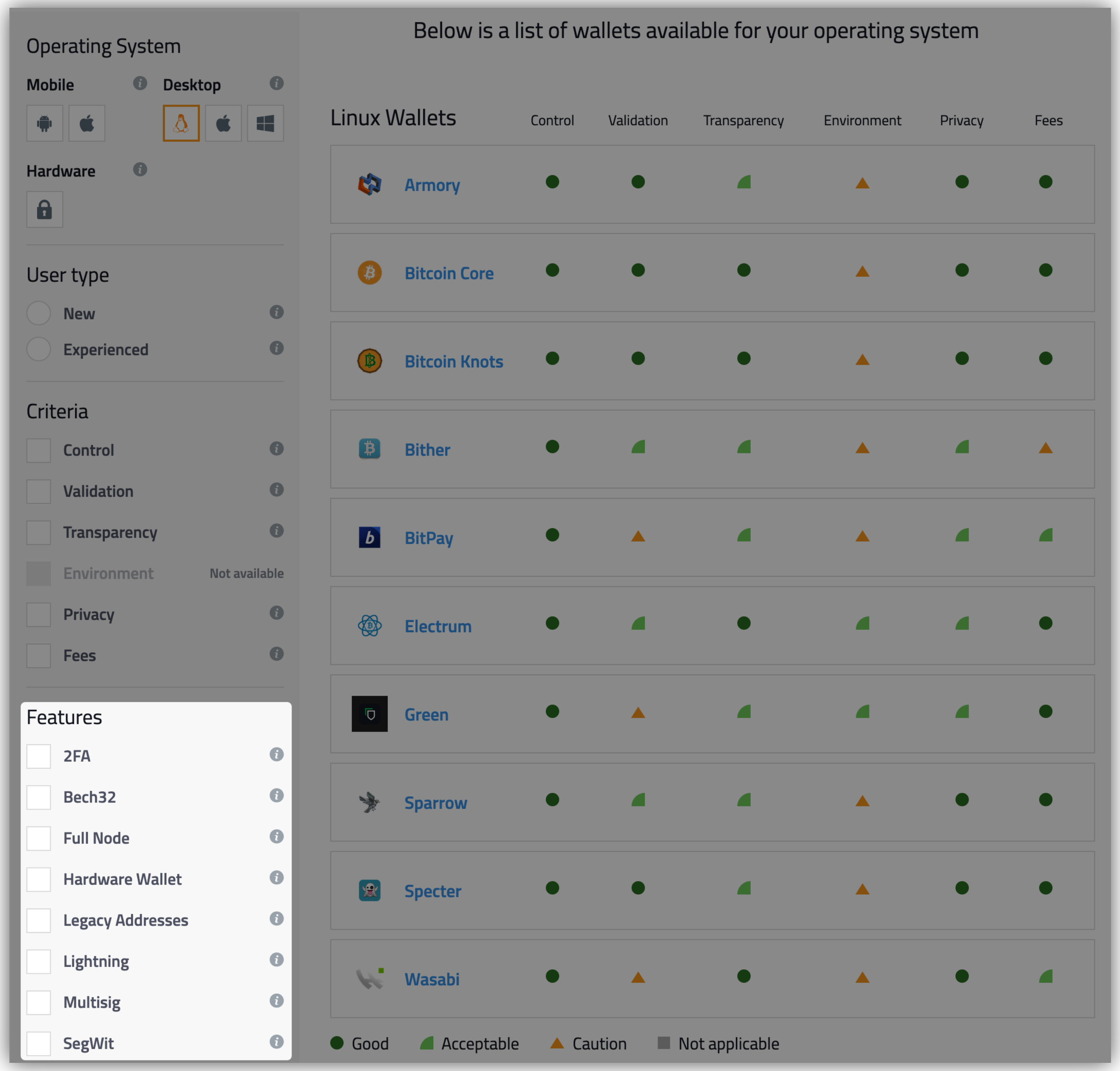
Roman Bögli, Alexander Boll, Alexander Schultheiß, and Timo Kehrer

Who rules bitcoin?

Source: bitcoin.org/en/choose-your-wallet

bitcoin





Operating System

Mobile ⓘ Desktop ⓘ

Hardware ⓘ

User type

○ New ⓘ

○ Experienced ⓘ

Criteria

□ Control ⓘ

□ Validation ⓘ

□ Transparency ⓘ

□ Environment Not available

□ Privacy ⓘ

□ Fees ⓘ

Features

□ 2FA ⓘ

□ Bech32 ⓘ

□ Full Node ⓘ

□ Hardware Wallet ⓘ

□ Legacy Addresses ⓘ

□ Lightning ⓘ

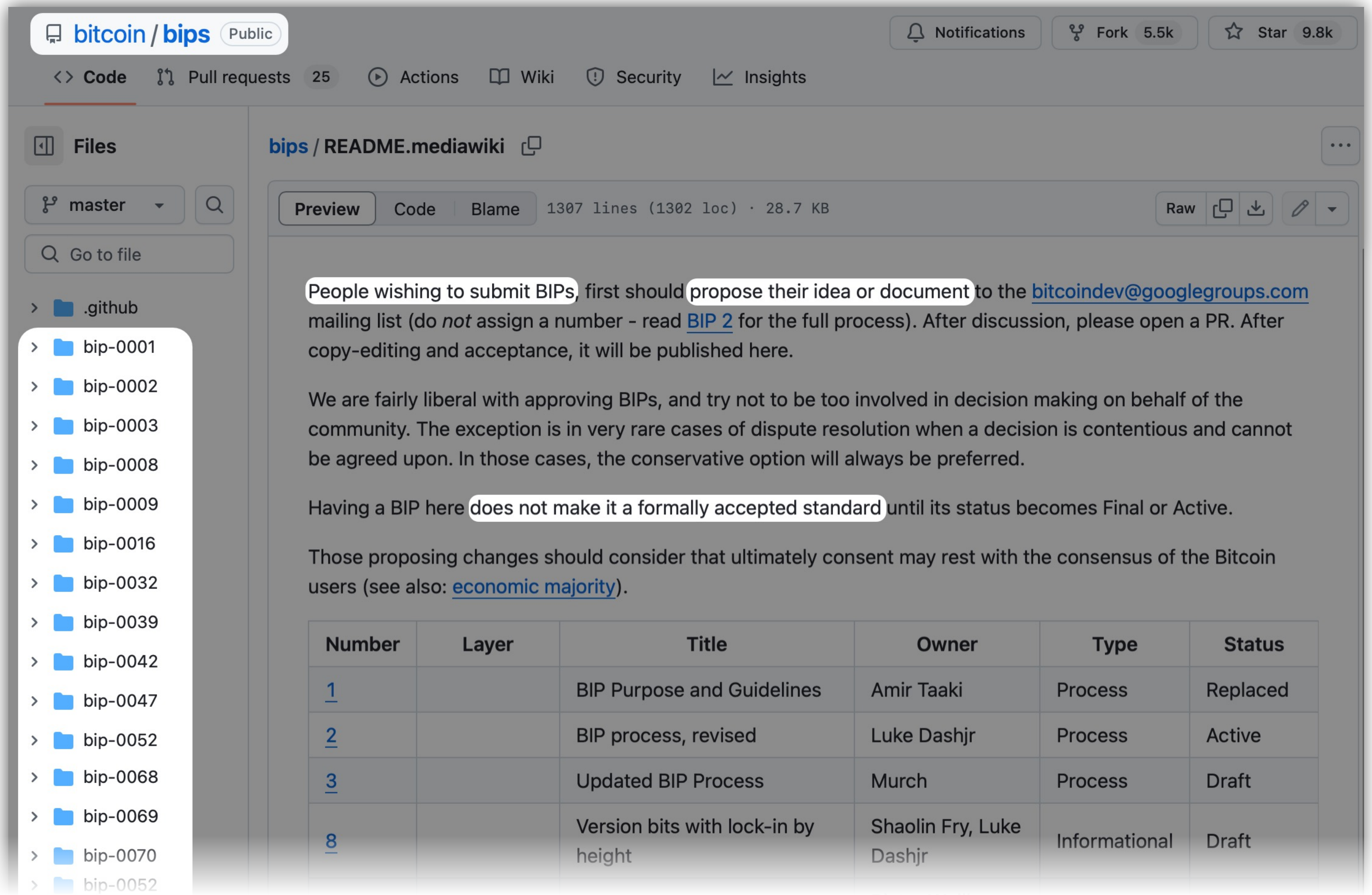
□ Multisig ⓘ

□ SegWit ⓘ

Below is a list of wallets available for your operating system

Linux Wallets	Control	Validation	Transparency	Environment	Privacy	Fees
Armory	●	●	◢	▲	●	●
Bitcoin Core	●	●	●	▲	●	●
Bitcoin Knots	●	●	●	▲	●	●
Bither	●	◢	◢	▲	◢	▲
BitPay	●	▲	◢	▲	◢	◢
Electrum	●	◢	●	◢	◢	●
Green	●	▲	◢	◢	◢	●
Sparrow	●	◢	◢	▲	●	●
Specter	●	●	◢	▲	●	●
Wasabi	●	▲	●	▲	●	◢

● Good ◢ Acceptable ▲ Caution ■ Not applicable



People wishing to submit BIPs, first should propose their idea or document to the bitcoindev@googlegroups.com mailing list (do *not* assign a number - read BIP 2 for the full process). After discussion, please open a PR. After copy-editing and acceptance, it will be published here.

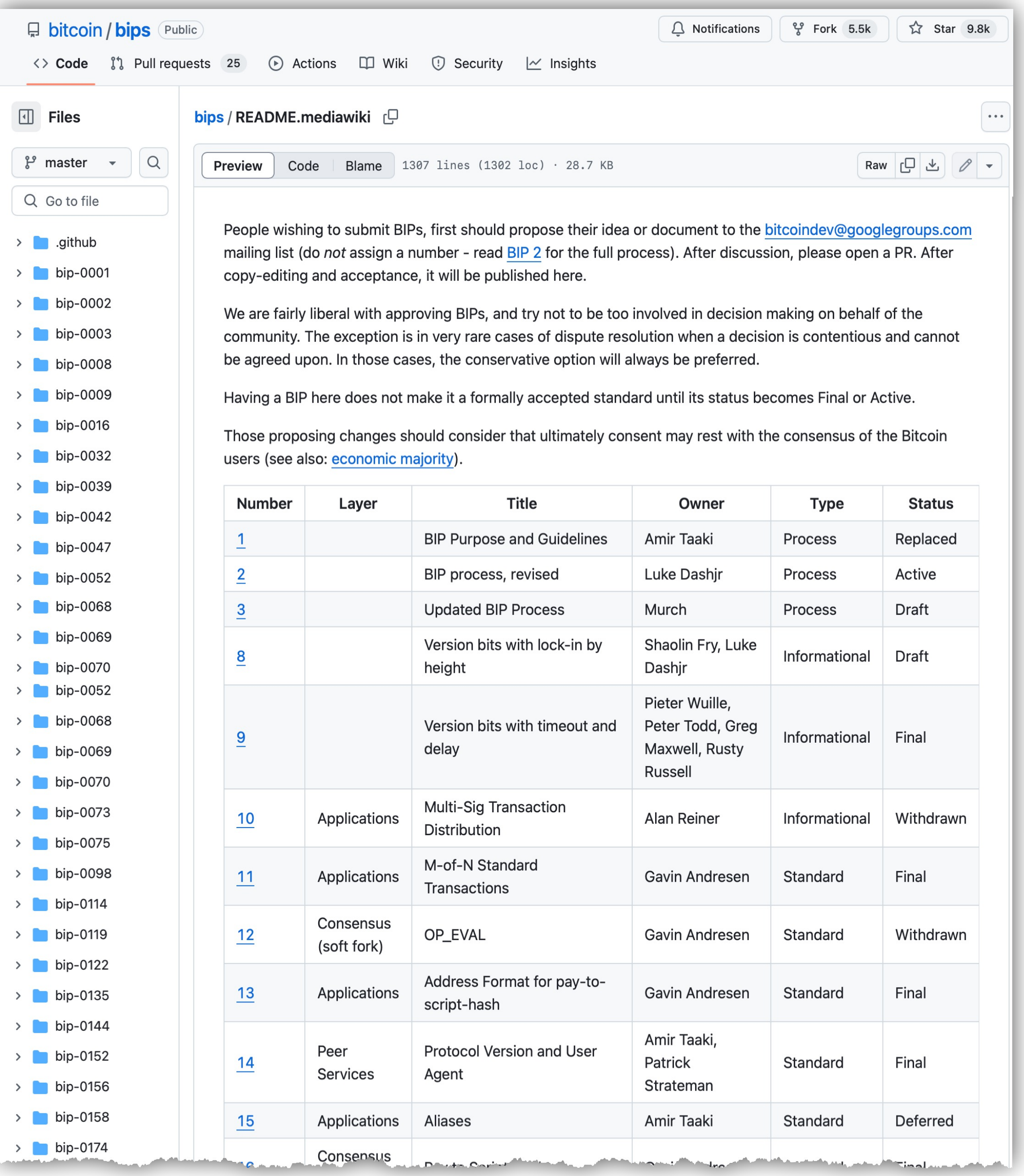
We are fairly liberal with approving BIPs, and try not to be too involved in decision making on behalf of the community. The exception is in very rare cases of dispute resolution when a decision is contentious and cannot be agreed upon. In those cases, the conservative option will always be preferred.

Having a BIP here does not make it a formally accepted standard until its status becomes Final or Active.

Those proposing changes should consider that ultimately consent may rest with the consensus of the Bitcoin users (see also: economic majority).

Number	Layer	Title	Owner	Type	Status
1		BIP Purpose and Guidelines	Amir Taaki	Process	Replaced
2		BIP process, revised	Luke Dashjr	Process	Active
3		Updated BIP Process	Murch	Process	Draft
8		Version bits with lock-in by height	Shaolin Fry, Luke Dashjr	Informational	Draft

Source: github.com/bitcoin/bips



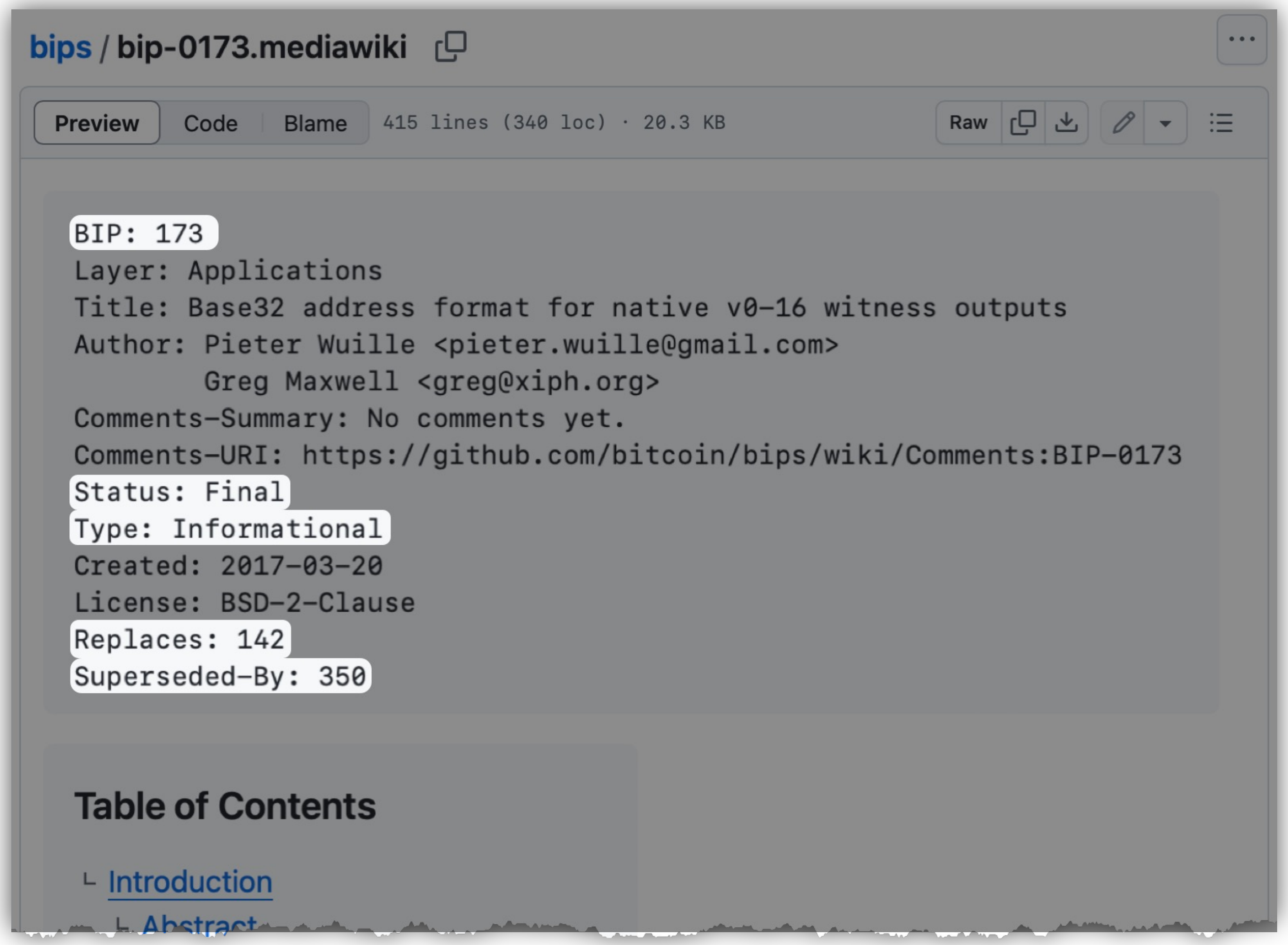
People wishing to submit BIPs, first should propose their idea or document to the bitcoindev@googlegroups.com mailing list (do *not* assign a number - read BIP 2 for the full process). After discussion, please open a PR. After copy-editing and acceptance, it will be published here.

We are fairly liberal with approving BIPs, and try not to be too involved in decision making on behalf of the community. The exception is in very rare cases of dispute resolution when a decision is contentious and cannot be agreed upon. In those cases, the conservative option will always be preferred.

Having a BIP here does not make it a formally accepted standard until its status becomes Final or Active.

Those proposing changes should consider that ultimately consent may rest with the consensus of the Bitcoin users (see also: economic majority).

Number	Layer	Title	Owner	Type	Status
1		BIP Purpose and Guidelines	Amir Taaki	Process	Replaced
2		BIP process, revised	Luke Dashjr	Process	Active
3		Updated BIP Process	Murch	Process	Draft
8		Version bits with lock-in by height	Shaolin Fry, Luke Dashjr	Informational	Draft
9		Version bits with timeout and delay	Pieter Wuille, Peter Todd, Greg Maxwell, Rusty Russell	Informational	Final
10	Applications	Multi-Sig Transaction Distribution	Alan Reiner	Informational	Withdrawn
11	Applications	M-of-N Standard Transactions	Gavin Andresen	Standard	Final
12	Consensus (soft fork)	OP_EVAL	Gavin Andresen	Standard	Withdrawn
13	Applications	Address Format for pay-to-script-hash	Gavin Andresen	Standard	Final
14	Peer Services	Protocol Version and User Agent	Amir Taaki, Patrick Strateman	Standard	Final
15	Applications	Aliases	Amir Taaki	Standard	Deferred
	Consensus				

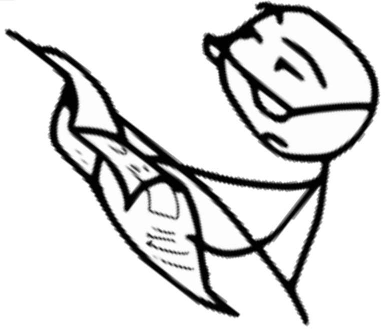


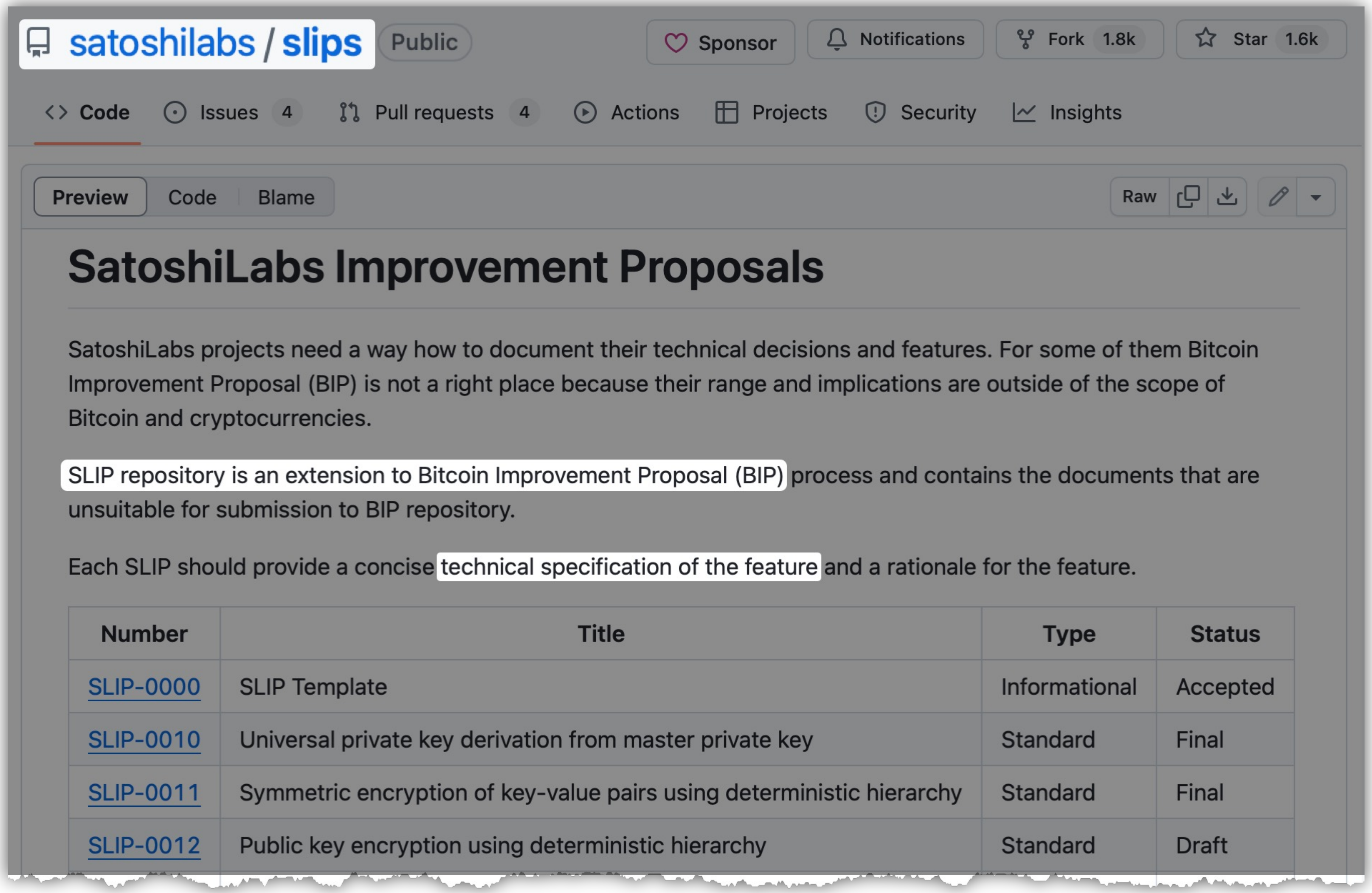
```
BIP: 173
Layer: Applications
Title: Base32 address format for native v0-16 witness outputs
Author: Pieter Wuille <pieter.wuille@gmail.com>
        Greg Maxwell <greg@xiph.org>
Comments-Summary: No comments yet.
Comments-URI: https://github.com/bitcoin/bips/wiki/Comments:BIP-0173
Status: Final
Type: Informational
Created: 2017-03-20
License: BSD-2-Clause
Replaces: 142
Superseded-By: 350
```

Table of Contents

- Introduction
  - Abstract

Source: github.com/bitcoin/bips/...bip-0173





SatoshiLabs Improvement Proposals

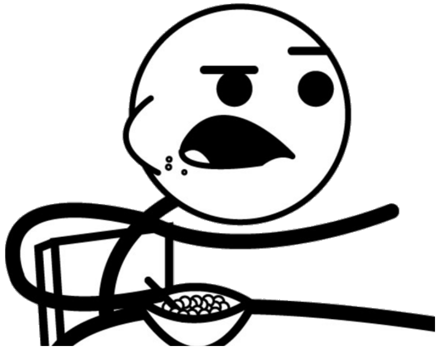
SatoshiLabs projects need a way how to document their technical decisions and features. For some of them Bitcoin Improvement Proposal (BIP) is not a right place because their range and implications are outside of the scope of Bitcoin and cryptocurrencies.

SLIP repository is an extension to Bitcoin Improvement Proposal (BIP) process and contains the documents that are unsuitable for submission to BIP repository.

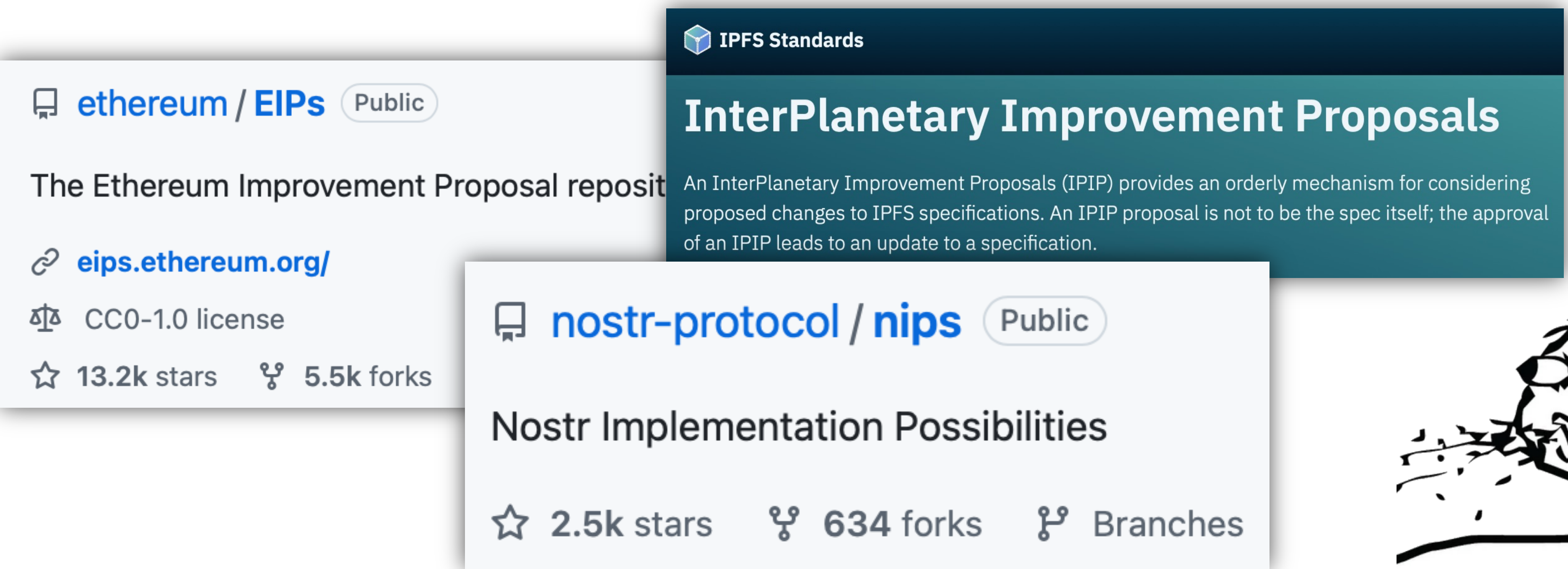
Each SLIP should provide a concise technical specification of the feature and a rationale for the feature.

Number	Title	Type	Status
SLIP-0000	SLIP Template	Informational	Accepted
SLIP-0010	Universal private key derivation from master private key	Standard	Final
SLIP-0011	Symmetric encryption of key-value pairs using deterministic hierarchy	Standard	Final
SLIP-0012	Public key encryption using deterministic hierarchy	Standard	Draft

Source: github.com/satoshilabs



bitcoin



ethereum / EIPs Public

The Ethereum Improvement Proposal reposit

eips.ethereum.org/

CC0-1.0 license

13.2k stars 5.5k forks

IPFS Standards

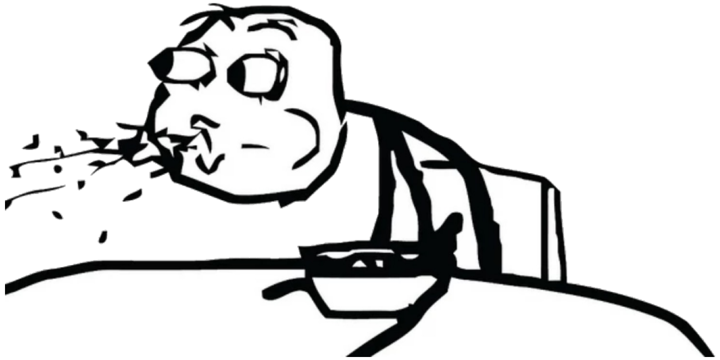
InterPlanetary Improvement Proposals

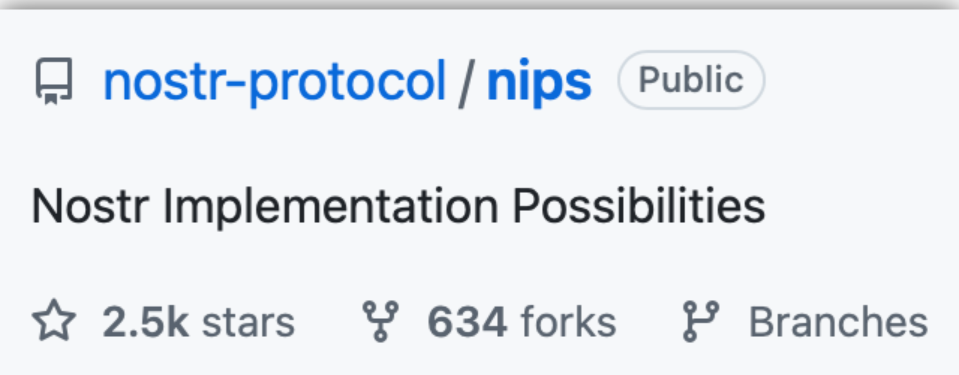
An InterPlanetary Improvement Proposals (IPIP) provides an orderly mechanism for considering proposed changes to IPFS specifications. An IPIP proposal is not to be the spec itself; the approval of an IPIP leads to an update to a specification.

nostr-protocol / nips Public

Nostr Implementation Possibilities

2.5k stars 634 forks Branches

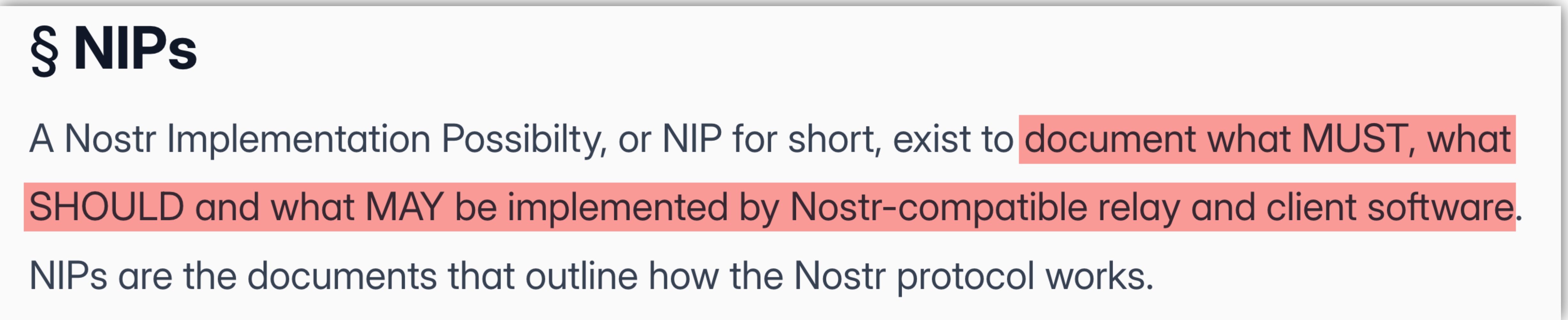




nostr-protocol / nips Public

Nostr Implementation Possibilities

☆ 2.5k stars 634 forks Branches

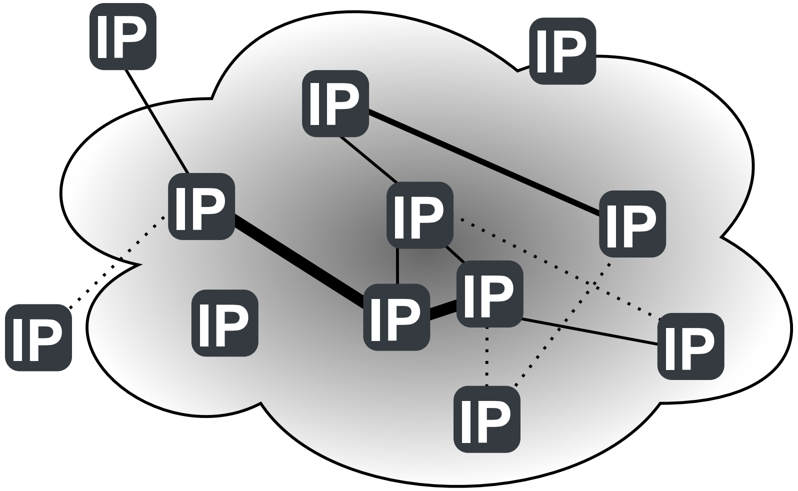


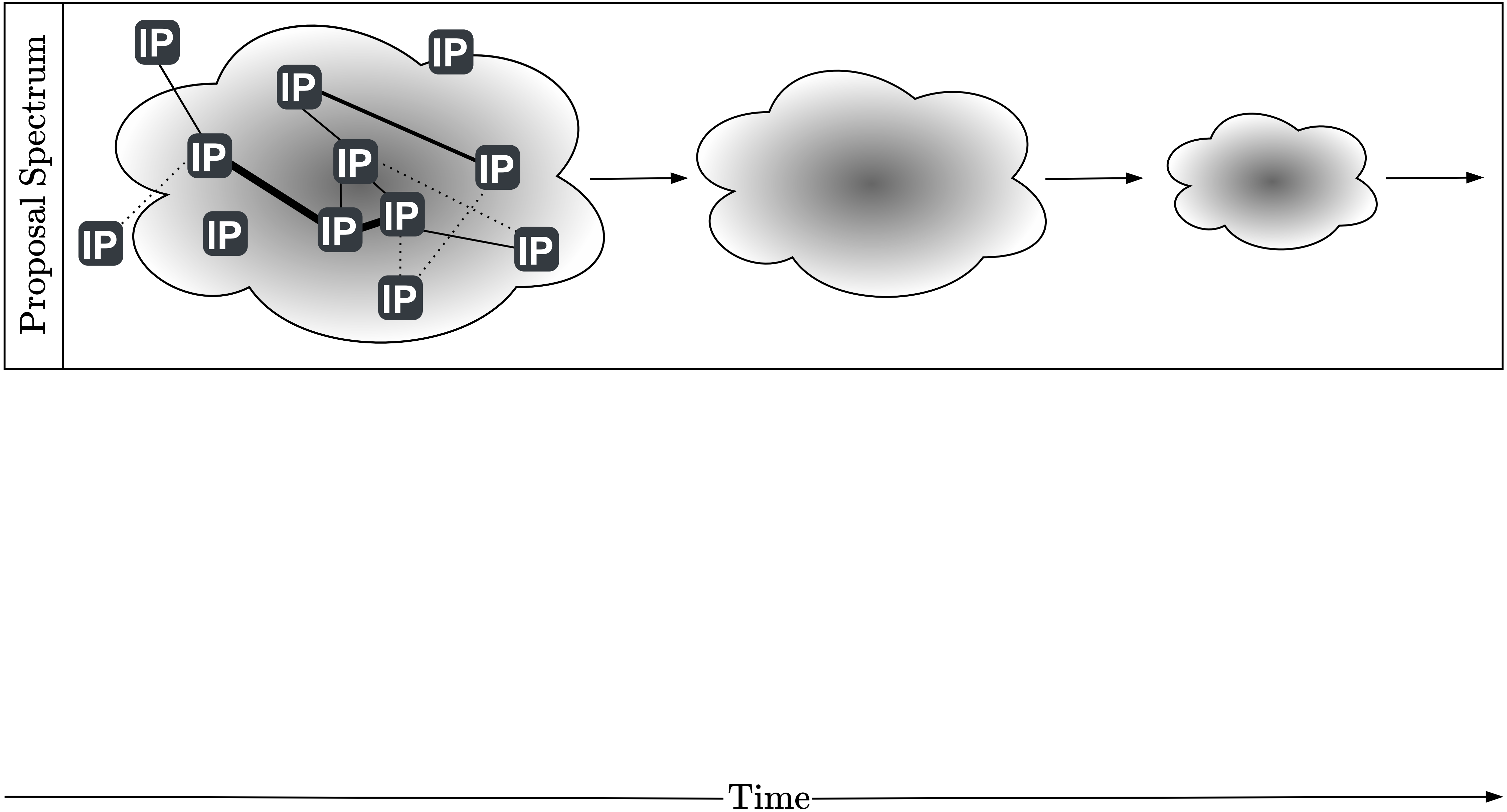
§ NIPs

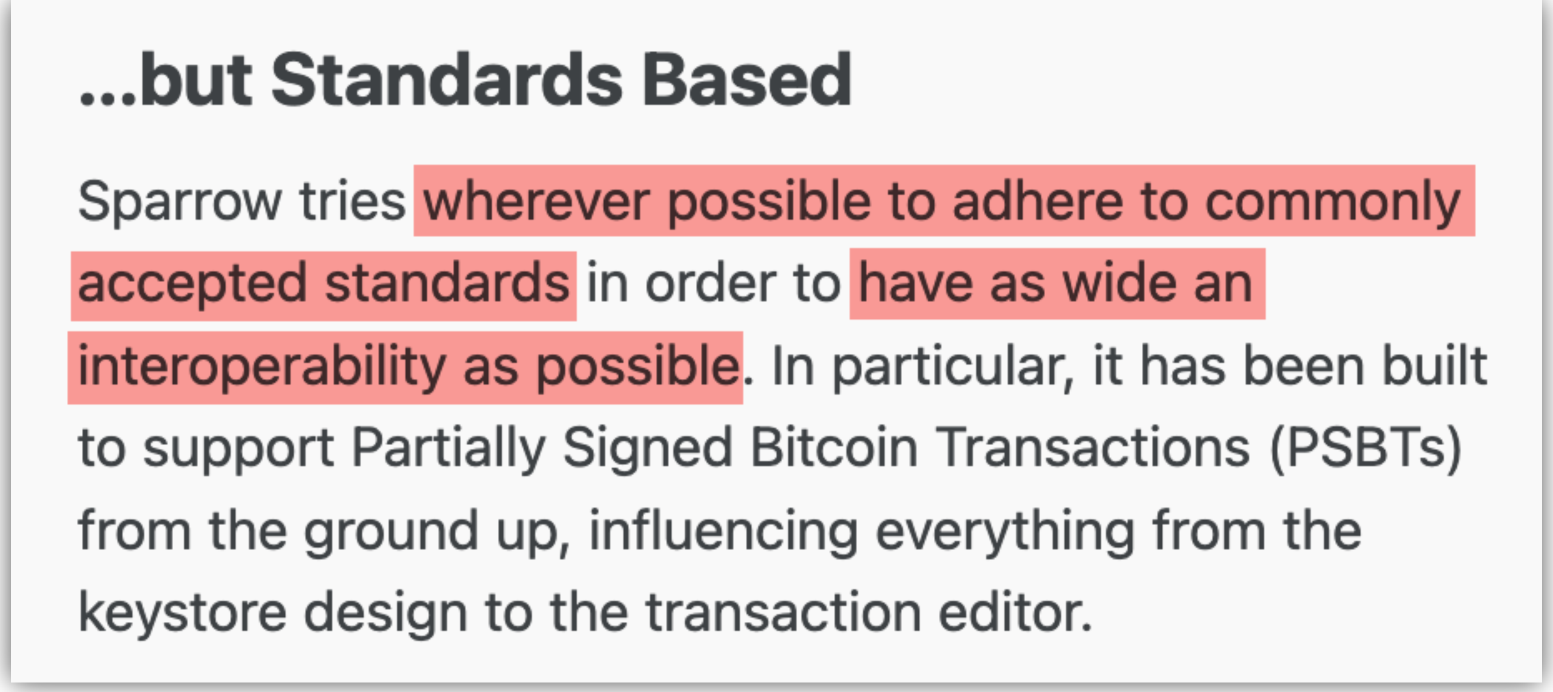
A Nostr Implementation Possibilty, or NIP for short, exist to document what MUST, what SHOULD and what MAY be implemented by Nostr-compatible relay and client software.

NIPs are the documents that outline how the Nostr protocol works.

Source: nostr.how


IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP

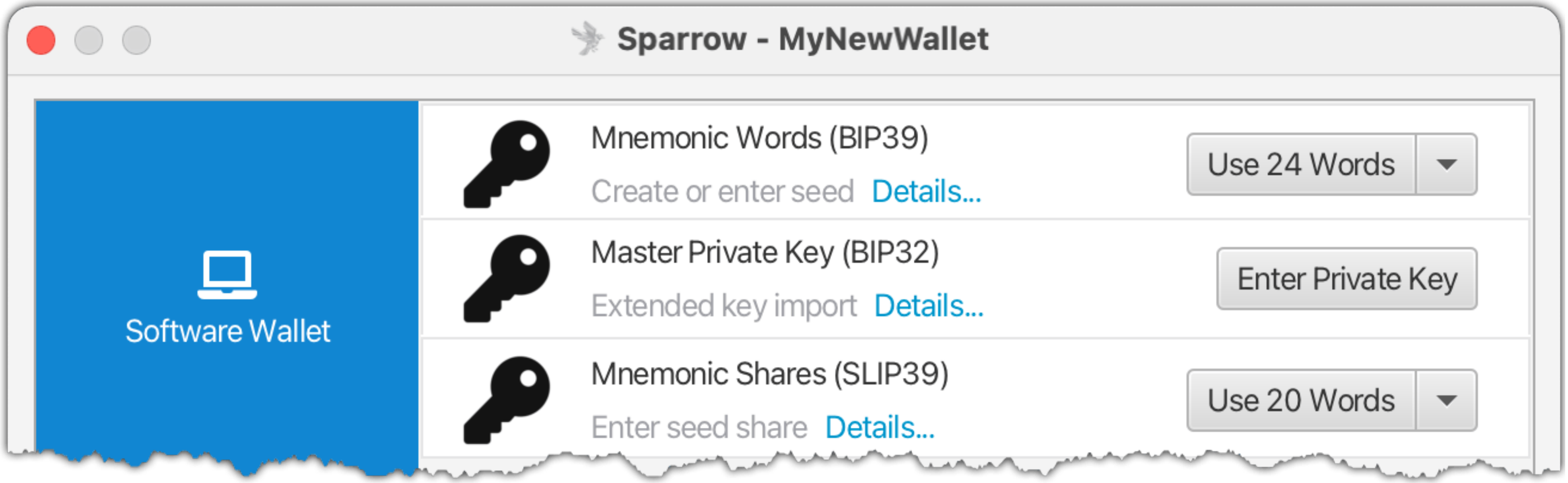

Proposal Spectrum
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
Time

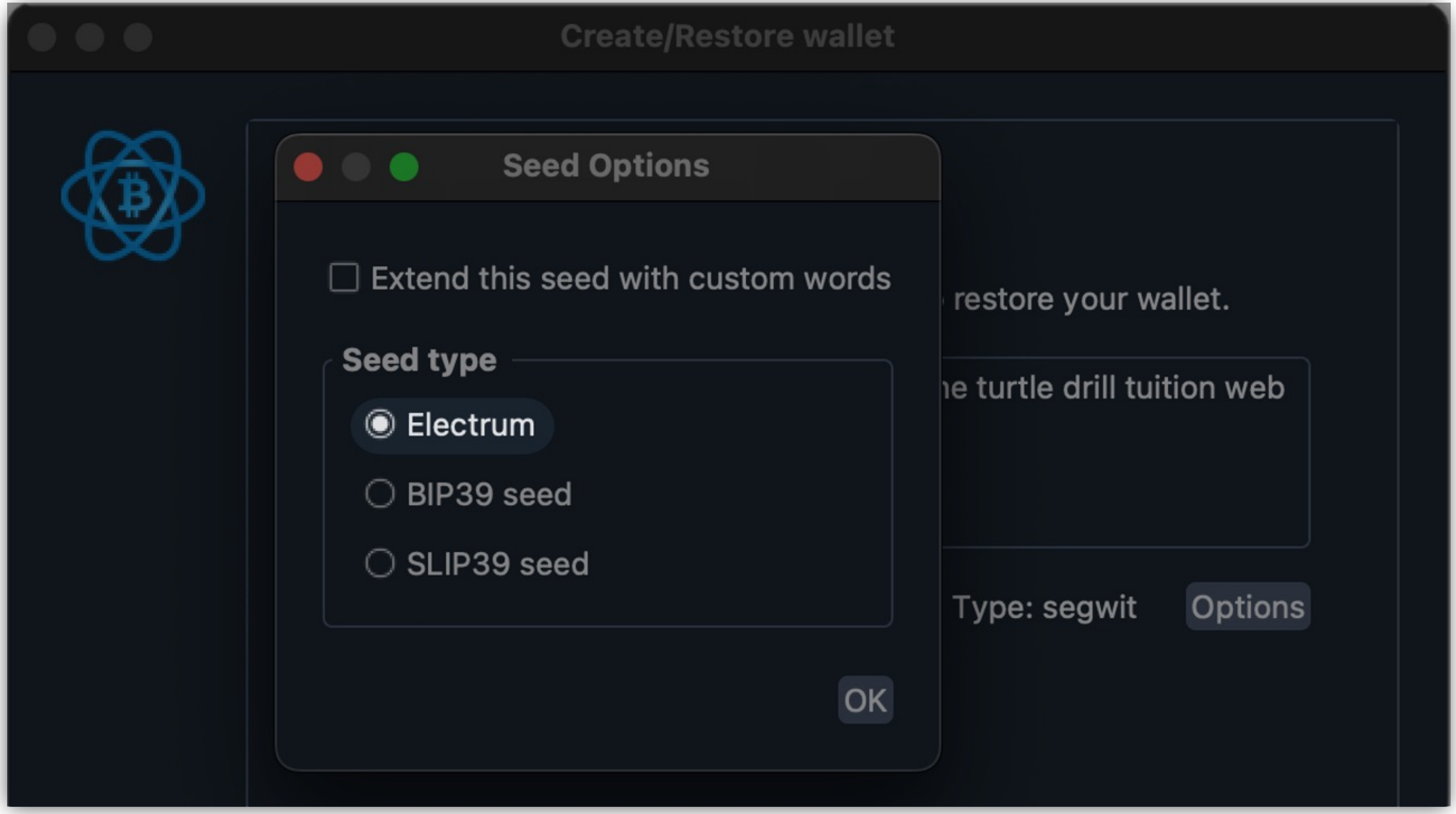


...but Standards Based

Sparrow tries wherever possible to adhere to commonly accepted standards in order to have as wide an interoperability as possible. In particular, it has been built to support Partially Signed Bitcoin Transactions (PSBTs) from the ground up, influencing everything from the keystore design to the transaction editor.

Source: sparrowwallet.com





Create/Restore wallet

Seed Options

☐ Extend this seed with custom words

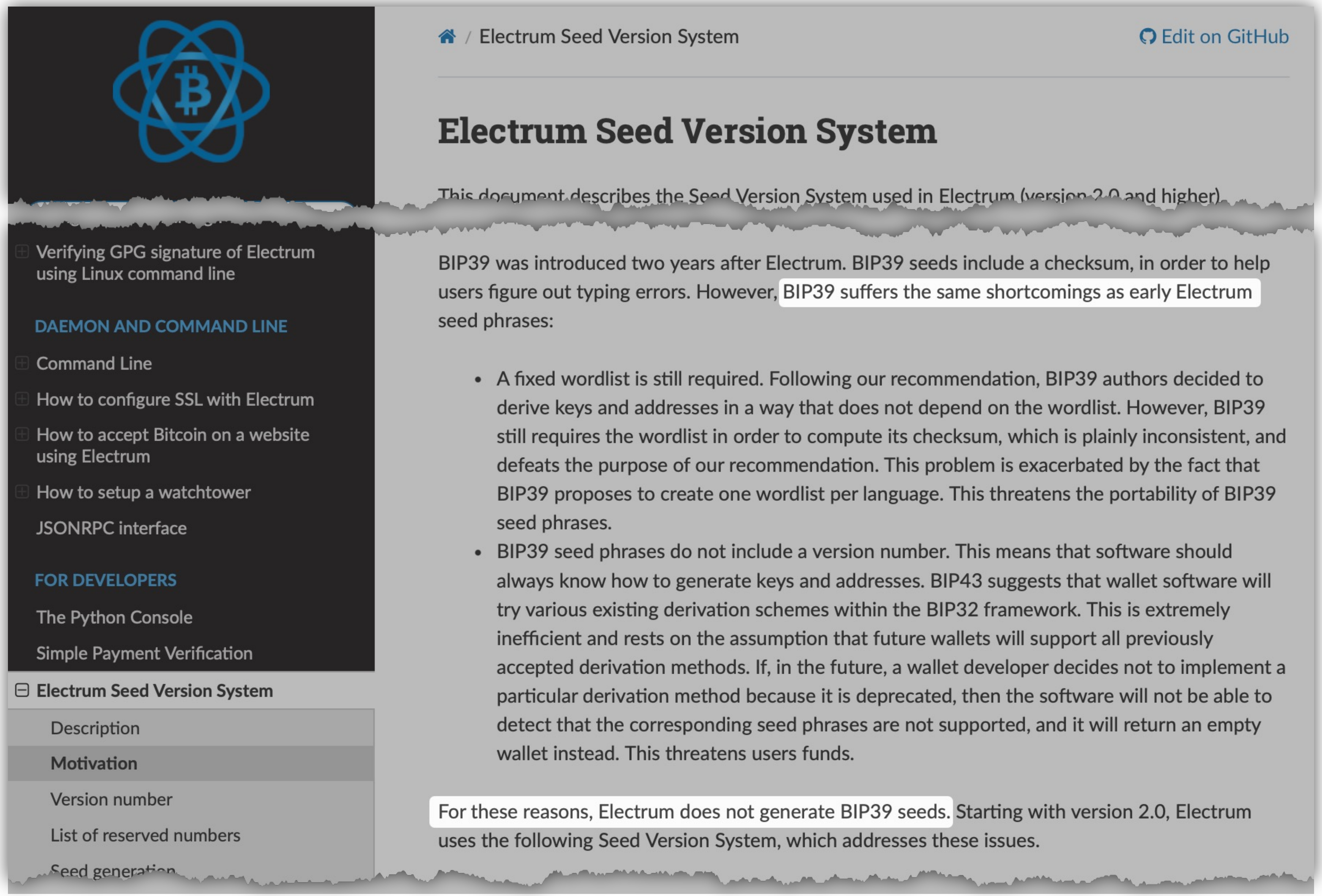
Seed type

● Electrum

○ BIP39 seed

○ SLIP39 seed

Source: electrum.readthedocs.io


Edit on GitHub
Verifying GPG signature of Electrum using Linux command line
DAEMON AND COMMAND LINE
Command Line
How to configure SSL with Electrum
How to accept Bitcoin on a website using Electrum
How to setup a watchtower
JSONRPC interface
FOR DEVELOPERS
The Python Console
Simple Payment Verification
Electrum Seed Version System
Description
Motivation
Version number
List of reserved numbers
Seed generation

/ Electrum Seed Version System

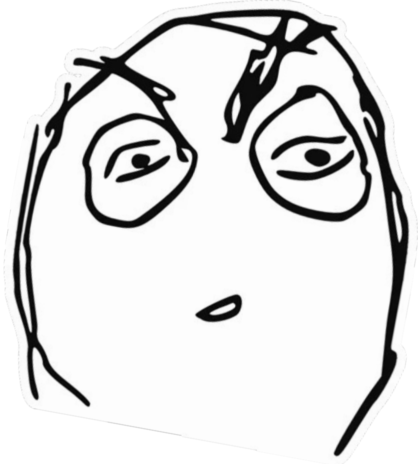
Electrum Seed Version System

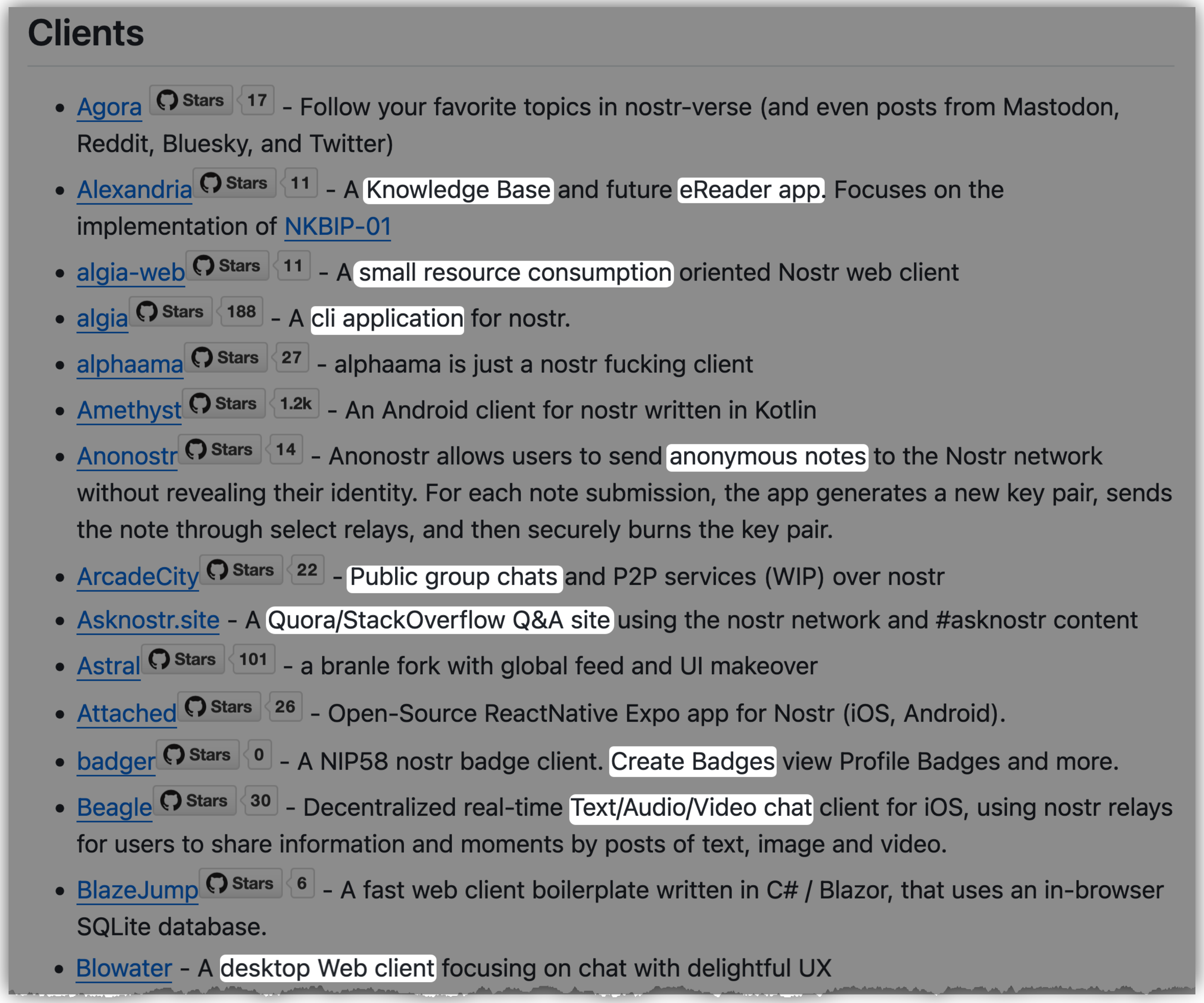
This document describes the Seed Version System used in Electrum (version 2.0 and higher).

BIP39 was introduced two years after Electrum. BIP39 seeds include a checksum, in order to help users figure out typing errors. However, BIP39 suffers the same shortcomings as early Electrum seed phrases:

- A fixed wordlist is still required. Following our recommendation, BIP39 authors decided to derive keys and addresses in a way that does not depend on the wordlist. However, BIP39 still requires the wordlist in order to compute its checksum, which is plainly inconsistent, and defeats the purpose of our recommendation. This problem is exacerbated by the fact that BIP39 proposes to create one wordlist per language. This threatens the portability of BIP39 seed phrases.
- BIP39 seed phrases do not include a version number. This means that software should always know how to generate keys and addresses. BIP43 suggests that wallet software will try various existing derivation schemes within the BIP32 framework. This is extremely inefficient and rests on the assumption that future wallets will support all previously accepted derivation methods. If, in the future, a wallet developer decides not to implement a particular derivation method because it is deprecated, then the software will not be able to detect that the corresponding seed phrases are not supported, and it will return an empty wallet instead. This threatens users funds.

For these reasons, Electrum does not generate BIP39 seeds. Starting with version 2.0, Electrum uses the following Seed Version System, which addresses these issues.

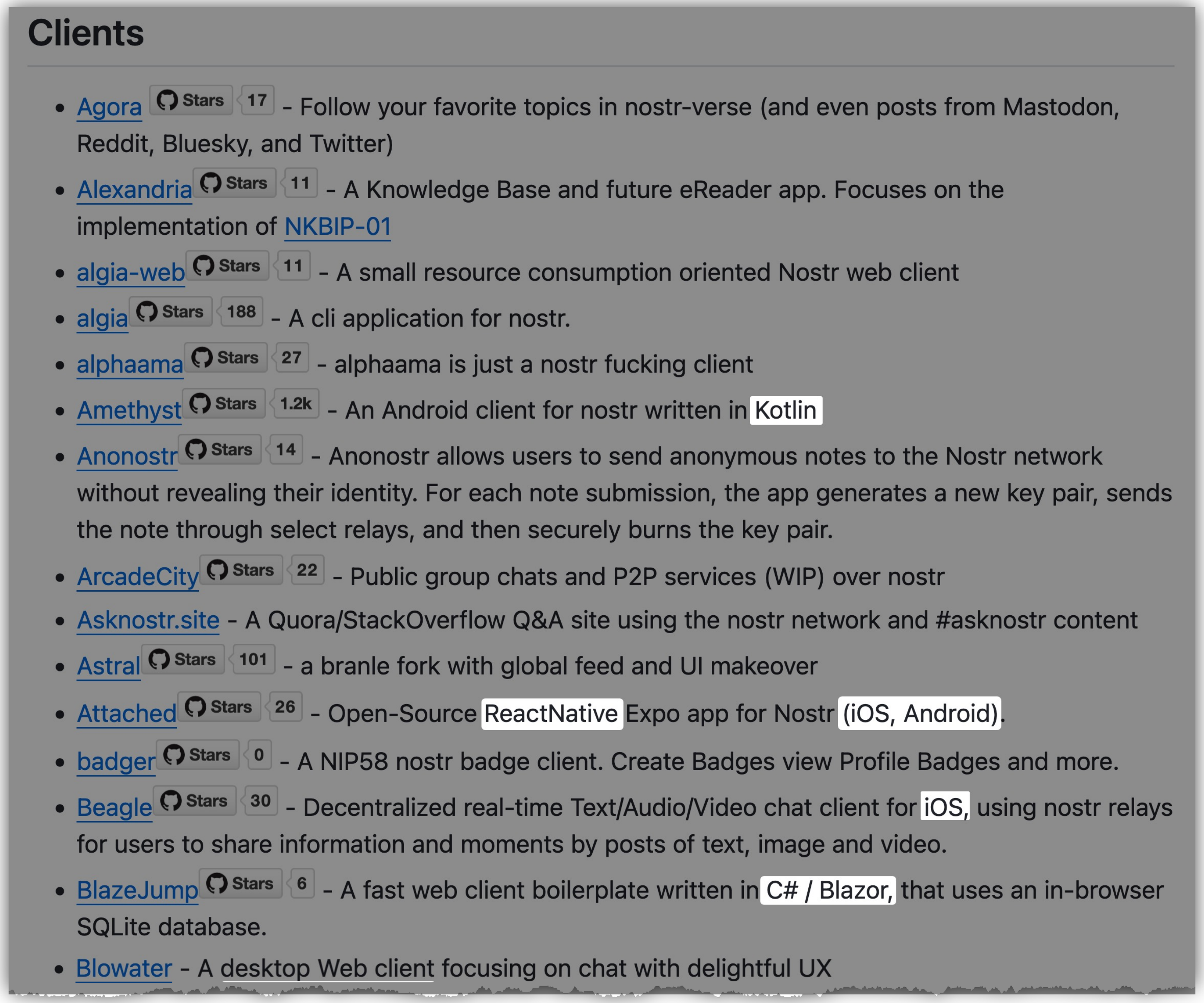




Clients

- Agora Stars 17 - Follow your favorite topics in nostr-verse (and even posts from Mastodon, Reddit, Bluesky, and Twitter)
- Alexandria Stars 11 - A Knowledge Base and future eReader app. Focuses on the implementation of NKBIP-01
- algia-web Stars 11 - A small resource consumption oriented Nostr web client
- algia Stars 188 - A cli application for nostr.
- alphaama Stars 27 - alphaama is just a nostr fucking client
- Amethyst Stars 1.2k - An Android client for nostr written in Kotlin
- Anonostr Stars 14 - Anonostr allows users to send anonymous notes to the Nostr network without revealing their identity. For each note submission, the app generates a new key pair, sends the note through select relays, and then securely burns the key pair.
- ArcadeCity Stars 22 - Public group chats and P2P services (WIP) over nostr
- Asknostr.site - A Quora/StackOverflow Q&A site using the nostr network and #asknostr content
- Astral Stars 101 - a branle fork with global feed and UI makeover
- Attached Stars 26 - Open-Source ReactNative Expo app for Nostr (iOS, Android).
- badger Stars 0 - A NIP58 nostr badge client. Create Badges view Profile Badges and more.
- Beagle Stars 30 - Decentralized real-time Text/Audio/Video chat client for iOS, using nostr relays for users to share information and moments by posts of text, image and video.
- BlazeJump Stars 6 - A fast web client boilerplate written in C# / Blazor, that uses an in-browser SQLite database.
- Blowater - A desktop Web client focusing on chat with delightful UX

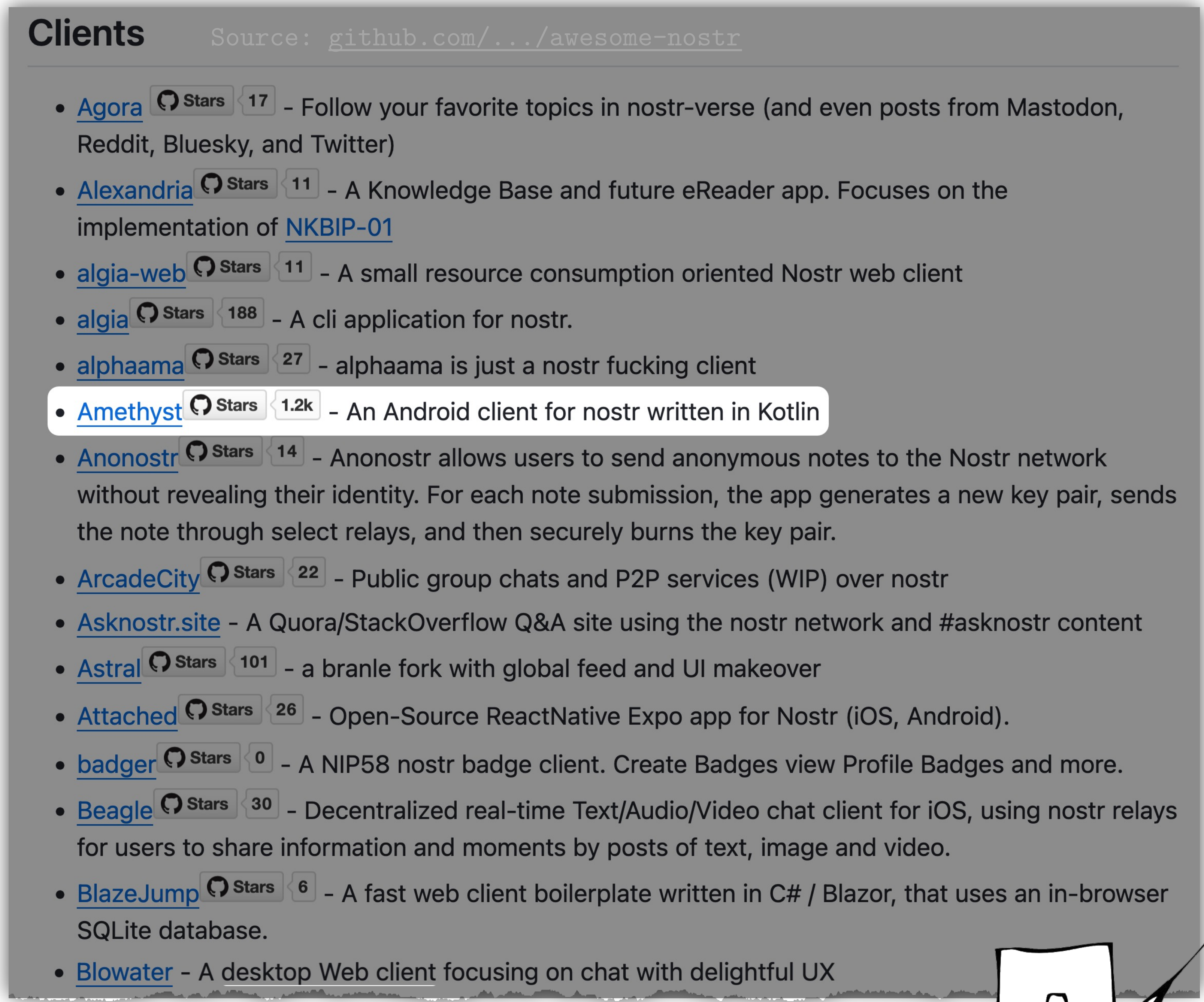
Source: github.com/.../awesome-nostr



Clients

- Agora Stars 17 - Follow your favorite topics in nostr-verse (and even posts from Mastodon, Reddit, Bluesky, and Twitter)
- Alexandria Stars 11 - A Knowledge Base and future eReader app. Focuses on the implementation of NKBIP-01
- algia-web Stars 11 - A small resource consumption oriented Nostr web client
- algia Stars 188 - A cli application for nostr.
- alphaama Stars 27 - alphaama is just a nostr fucking client
- Amethyst Stars 1.2k - An Android client for nostr written in Kotlin
- Anonostr Stars 14 - Anonostr allows users to send anonymous notes to the Nostr network without revealing their identity. For each note submission, the app generates a new key pair, sends the note through select relays, and then securely burns the key pair.
- ArcadeCity Stars 22 - Public group chats and P2P services (WIP) over nostr
- Asknostr.site - A Quora/StackOverflow Q&A site using the nostr network and #asknostr content
- Astral Stars 101 - a branle fork with global feed and UI makeover
- Attached Stars 26 - Open-Source ReactNative Expo app for Nostr (iOS, Android).
- badger Stars 0 - A NIP58 nostr badge client. Create Badges view Profile Badges and more.
- Beagle Stars 30 - Decentralized real-time Text/Audio/Video chat client for iOS, using nostr relays for users to share information and moments by posts of text, image and video.
- BlazeJump Stars 6 - A fast web client boilerplate written in C# / Blazor, that uses an in-browser SQLite database.
- Blowater - A desktop Web client focusing on chat with delightful UX

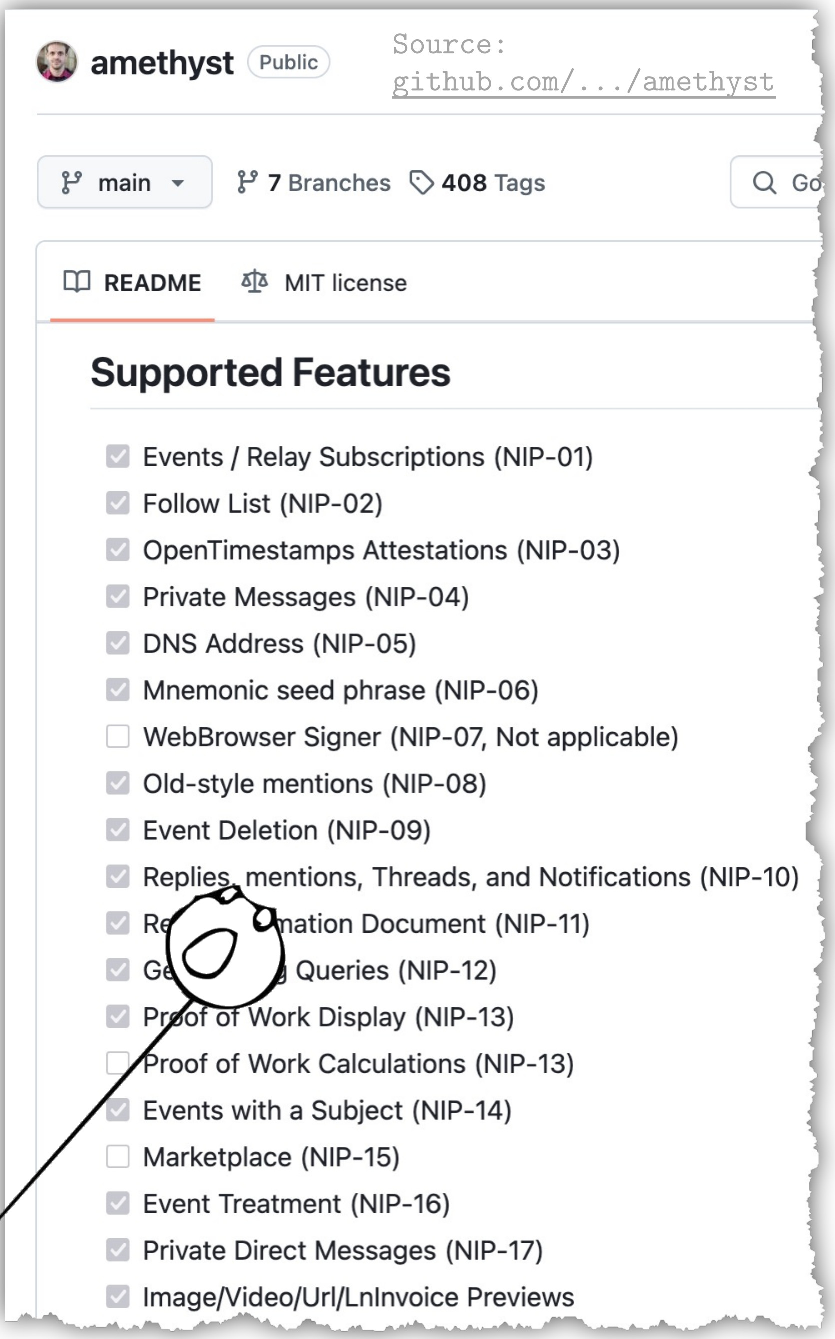
Source: github.com/.../awesome-nostr



Clients

Source: github.com/.../awesome-nostr

- Agora Stars 17 - Follow your favorite topics in nostr-verse (and even posts from Mastodon, Reddit, Bluesky, and Twitter)
- Alexandria Stars 11 - A Knowledge Base and future eReader app. Focuses on the implementation of NKBIP-01
- algia-web Stars 11 - A small resource consumption oriented Nostr web client
- algia Stars 188 - A cli application for nostr.
- alphaama Stars 27 - alphaama is just a nostr fucking client
- Amethyst Stars 1.2k - An Android client for nostr written in Kotlin
- Anonostr Stars 14 - Anonostr allows users to send anonymous notes to the Nostr network without revealing their identity. For each note submission, the app generates a new key pair, sends the note through select relays, and then securely burns the key pair.
- ArcadeCity Stars 22 - Public group chats and P2P services (WIP) over nostr
- Asknostr.site - A Quora/StackOverflow Q&A site using the nostr network and #asknostr content
- Astral Stars 101 - a branle fork with global feed and UI makeover
- Attached Stars 26 - Open-Source ReactNative Expo app for Nostr (iOS, Android).
- badger Stars 0 - A NIP58 nostr badge client. Create Badges view Profile Badges and more.
- Beagle Stars 30 - Decentralized real-time Text/Audio/Video chat client for iOS, using nostr relays for users to share information and moments by posts of text, image and video.
- BlazeJump Stars 6 - A fast web client boilerplate written in C# / Blazor, that uses an in-browser SQLite database.
- Blowater - A desktop Web client focusing on chat with delightful UX



amethyst Public

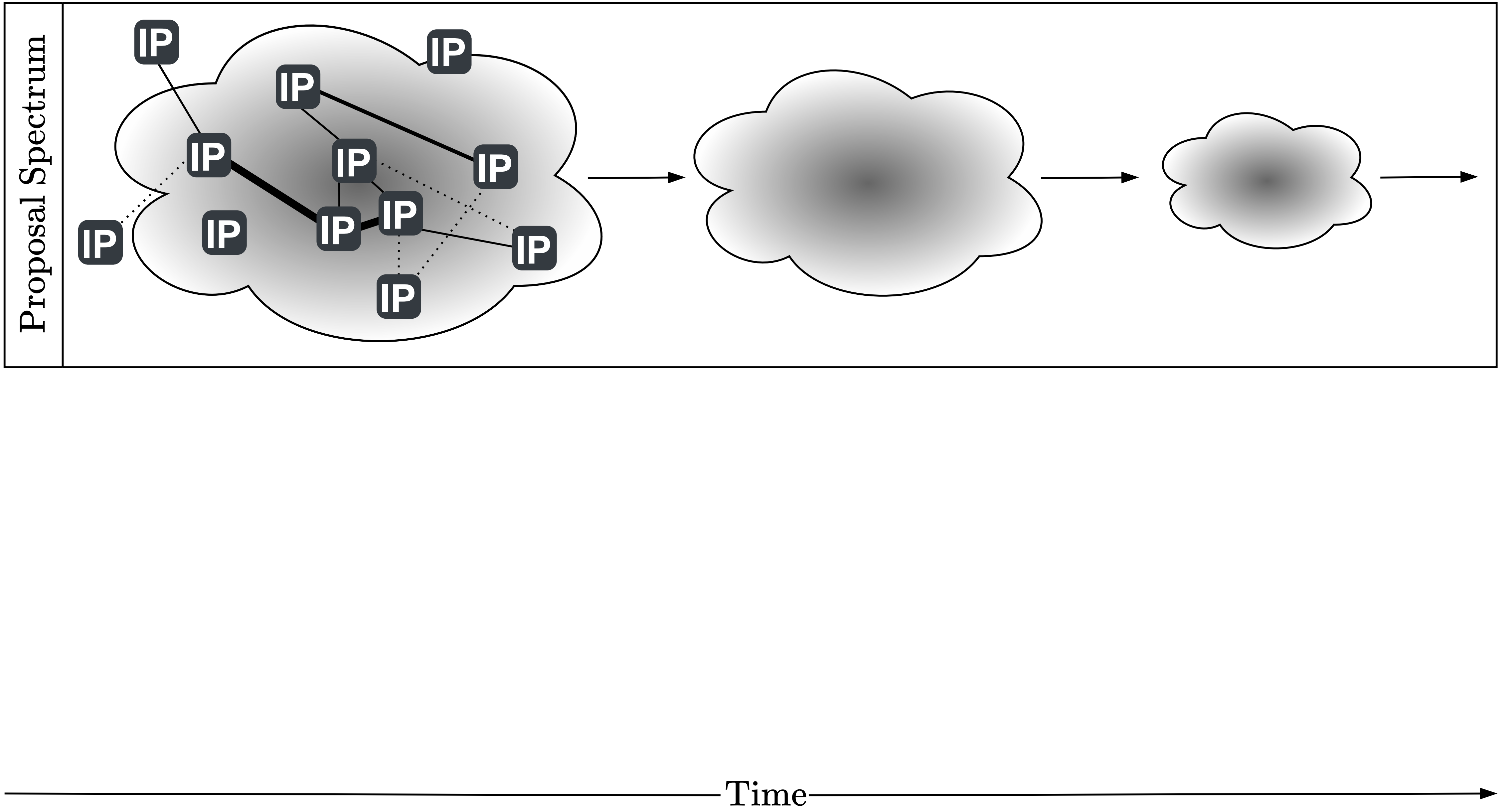
Source: github.com/.../amethyst

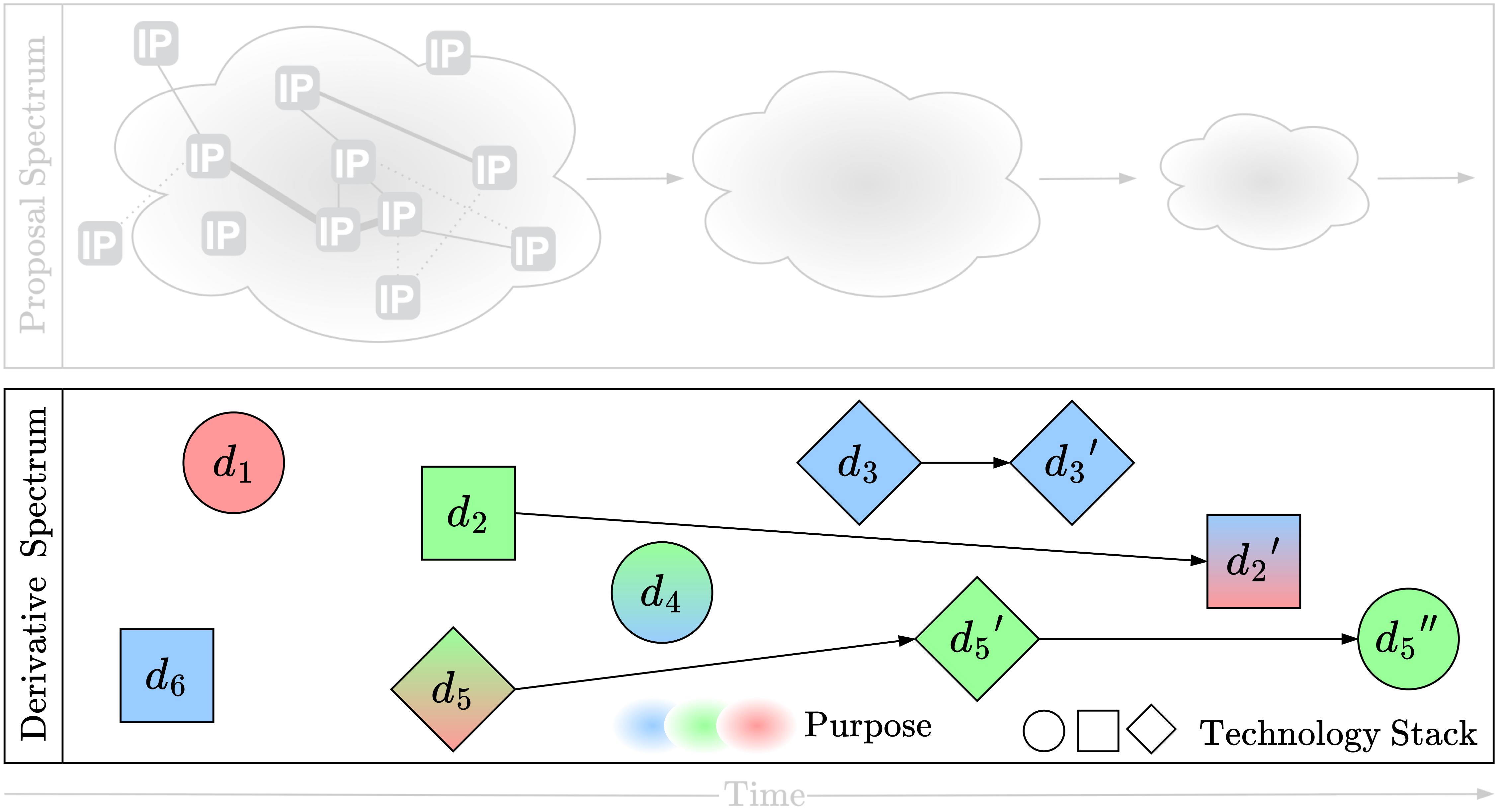
main 7 Branches 408 Tags

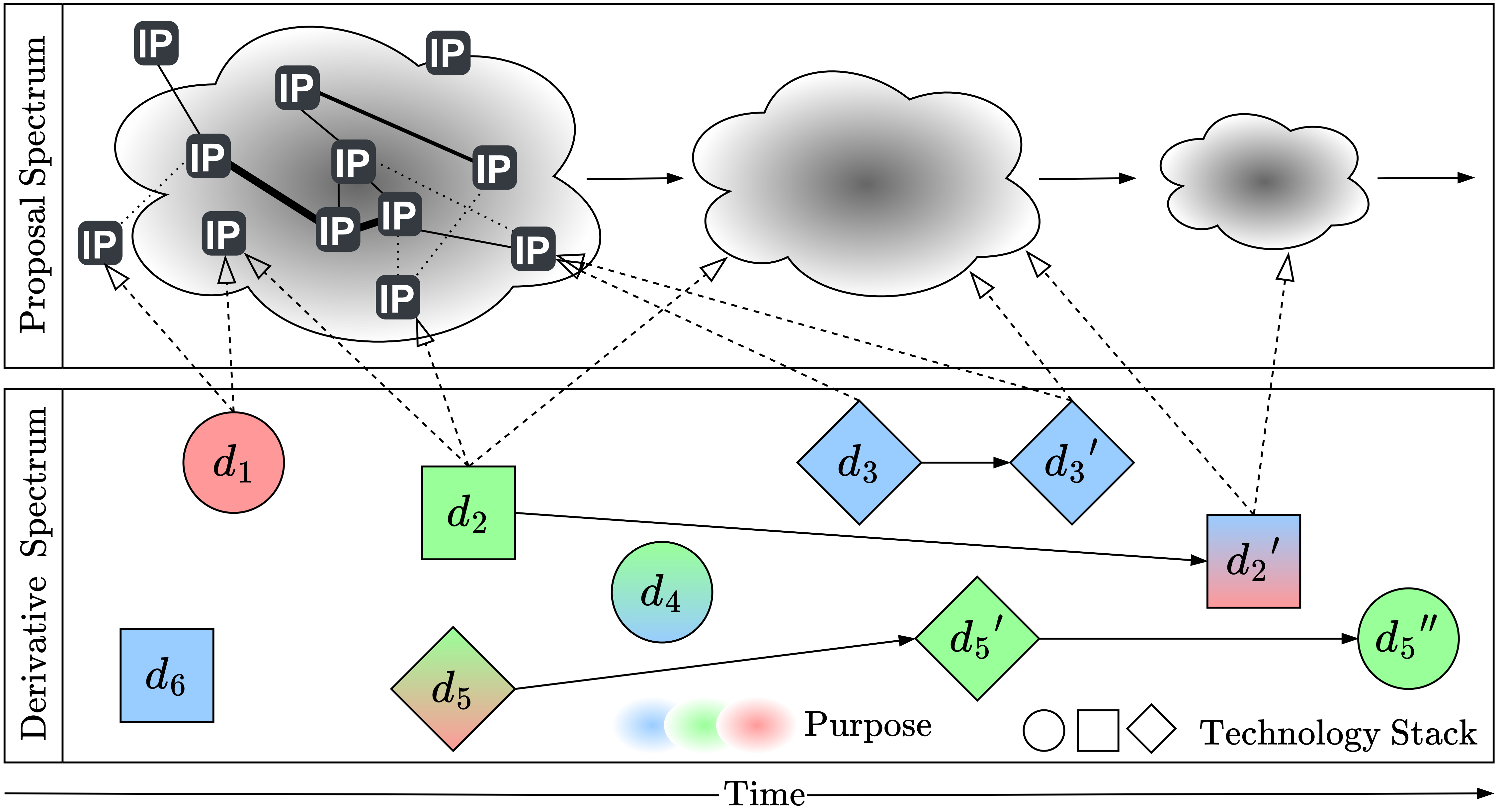
README MIT license

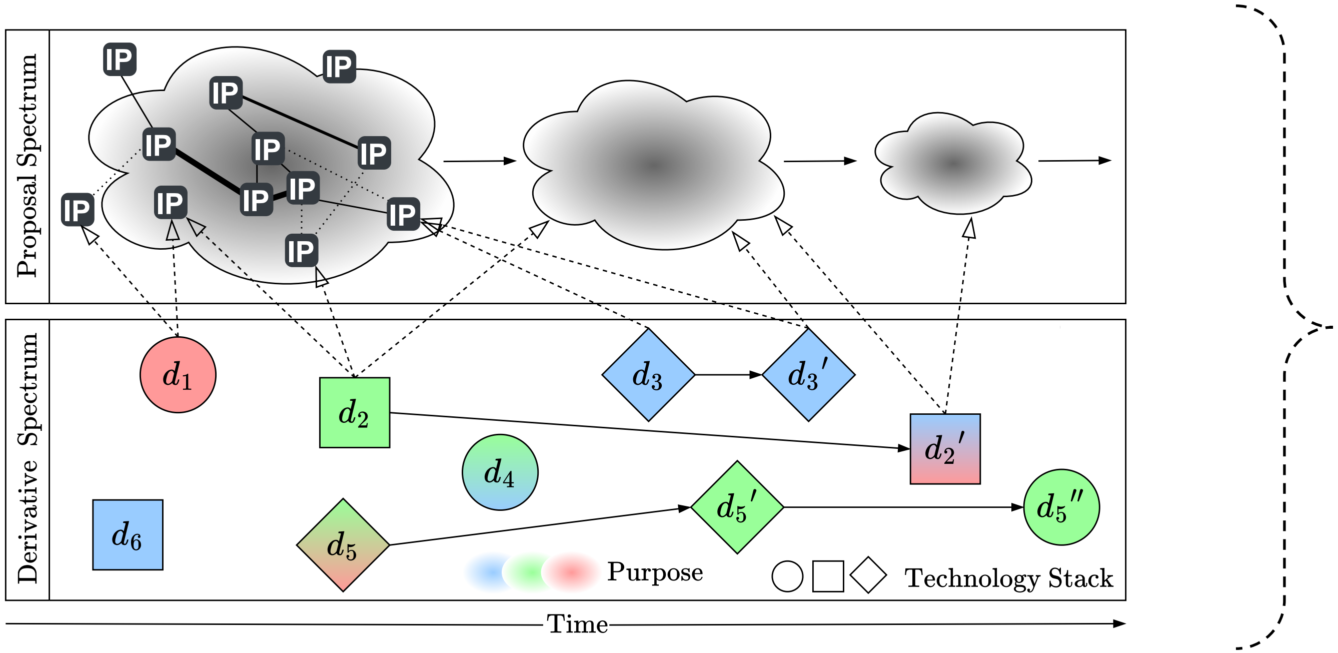
Supported Features

- ☑ Events / Relay Subscriptions (NIP-01)
- ☑ Follow List (NIP-02)
- ☑ OpenTimestamps Attestations (NIP-03)
- ☑ Private Messages (NIP-04)
- ☑ DNS Address (NIP-05)
- ☑ Mnemonic seed phrase (NIP-06)
- ☐ WebBrowser Signer (NIP-07, Not applicable)
- ☑ Old-style mentions (NIP-08)
- ☑ Event Deletion (NIP-09)
- ☑ Replies, mentions, Threads, and Notifications (NIP-10)
- ☑ Re mation Document (NIP-11)
- ☑ Ge g Queries (NIP-12)
- ☑ Proof of Work Display (NIP-13)
- ☐ Proof of Work Calculations (NIP-13)
- ☑ Events with a Subject (NIP-14)
- ☐ Marketplace (NIP-15)
- ☑ Event Treatment (NIP-16)
- ☑ Private Direct Messages (NIP-17)
- ☑ Image/Video/Url/LnInvoice Previews


Proposal Spectrum
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
Time


Proposal Spectrum
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
IP
Derivative Spectrum
d_1
d_2
d_3
$d_3{}'$
$d_2{}'$
d_4
d_6
d_5
$d_5{}'$
$d_5{}''$
Purpose
Technology Stack
Time


Proposal Spectrum
IP
Derivative Spectrum
d_1
d_2
d_3
d_3'
d_2'
d_4
d_5
d_5'
d_5''
d_6
Purpose
○ □ ◇ Technology Stack
Time



Community-Driven Variability (CDV)

Community-Driven Variability (CDV)

- C1 Crowdsourcing
- C2 Improvement Proposals
- C3 Independent Derivatives
- C4 Interoperability
- C5 Decoupled Evolution

So what?

P1 Missing overview in proposal spectrum

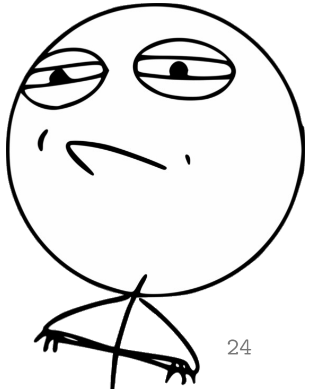
P2 Missing overview in derivative spectrum

P3 IP change impact assessment

P4 Misalignment of proposal & derivative spectrum

P5 Level of derivative interoperability

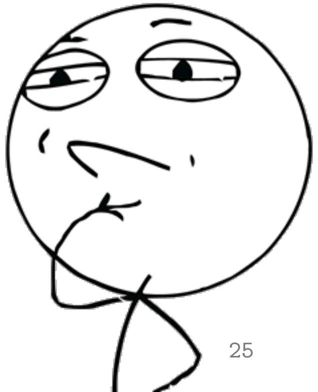
P6 Ecosystem forks

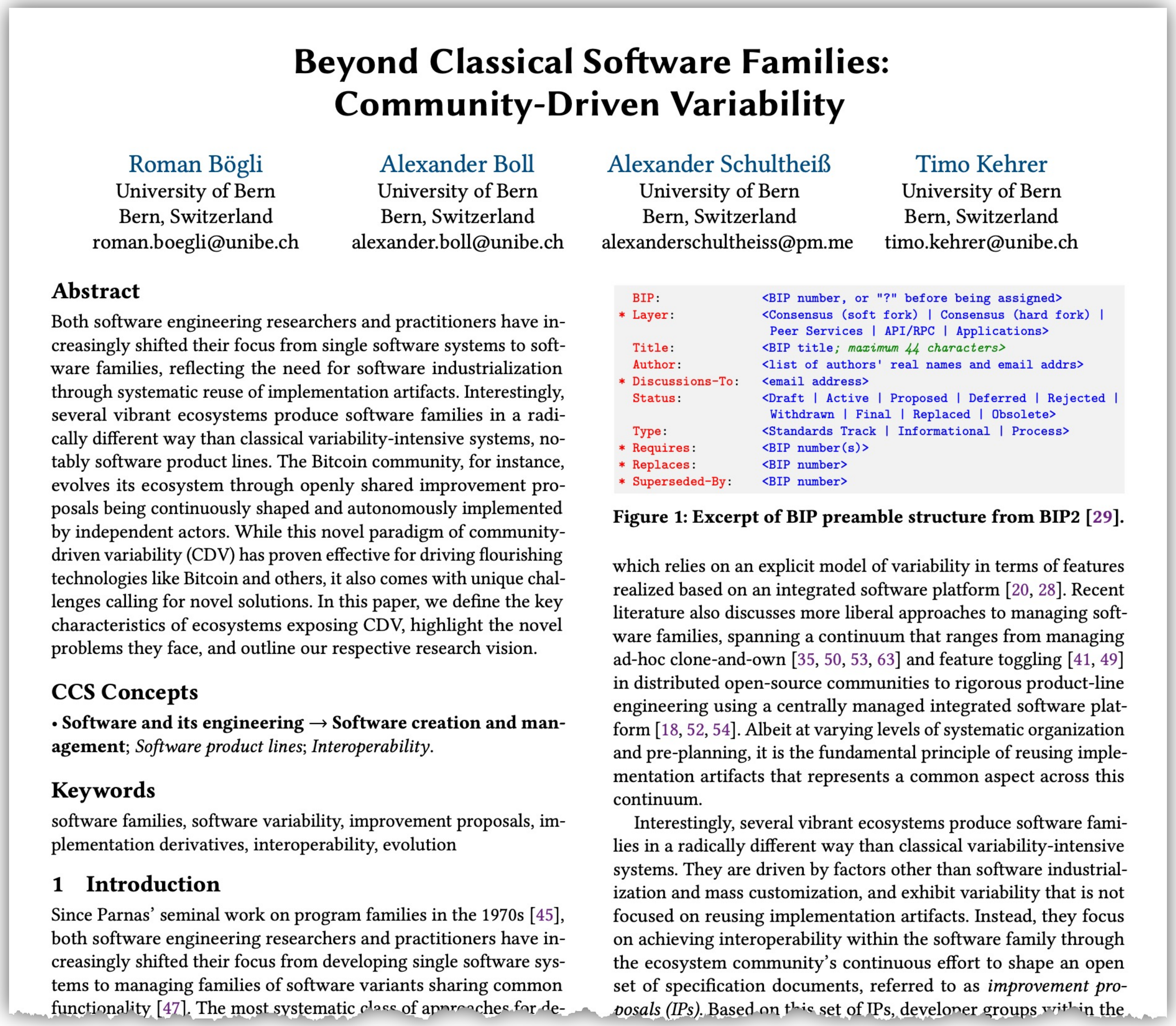


RG1 Systematic treatment of CDV in proposal spectrum

RG2 Supporting cohesive evolution of proposal and derivative spectrum

RG3 Methodical handling of derivative interoperability impairment





Beyond Classical Software Families: Community-Driven Variability

Roman Bögli
University of Bern
Bern, Switzerland
roman.boegli@unibe.ch

Alexander Boll
University of Bern
Bern, Switzerland
alexander.boll@unibe.ch

Alexander Schultheiß
University of Bern
Bern, Switzerland
alexanderschultheiss@pm.me

Timo Kehrer
University of Bern
Bern, Switzerland
timo.kehrer@unibe.ch

Abstract

Both software engineering researchers and practitioners have increasingly shifted their focus from single software systems to software families, reflecting the need for software industrialization through systematic reuse of implementation artifacts. Interestingly, several vibrant ecosystems produce software families in a radically different way than classical variability-intensive systems, notably software product lines. The Bitcoin community, for instance, evolves its ecosystem through openly shared improvement proposals being continuously shaped and autonomously implemented by independent actors. While this novel paradigm of community-driven variability (CDV) has proven effective for driving flourishing technologies like Bitcoin and others, it also comes with unique challenges calling for novel solutions. In this paper, we define the key characteristics of ecosystems exposing CDV, highlight the novel problems they face, and outline our respective research vision.

CCS Concepts

• **Software and its engineering → Software creation and management**; *Software product lines*; *Interoperability*.

Keywords

software families, software variability, improvement proposals, implementation derivatives, interoperability, evolution

1 Introduction

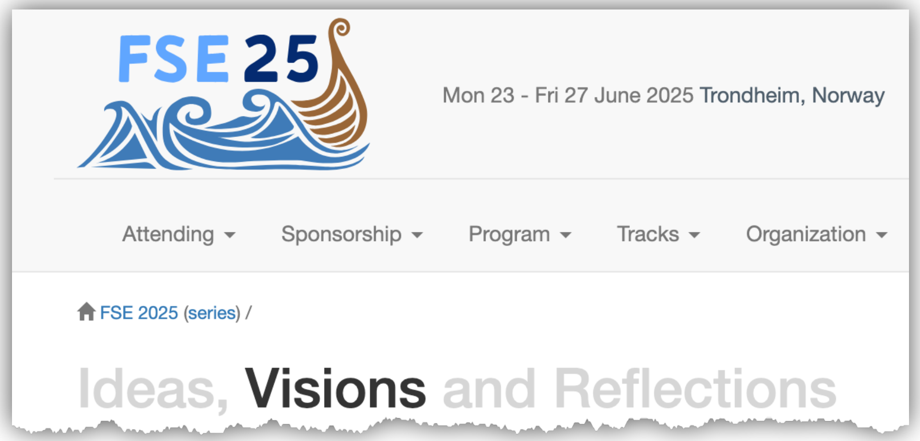
Since Parnas' seminal work on program families in the 1970s [45], both software engineering researchers and practitioners have increasingly shifted their focus from developing single software systems to managing families of software variants sharing common functionality [47]. The most systematic class of approaches for de-

```
  BIP:              <BIP number, or "?" before being assigned>
* Layer:            <Consensus (soft fork) | Consensus (hard fork) |
                     Peer Services | API/RPC | Applications>
  Title:            <BIP title; maximum 44 characters>
  Author:           <list of authors' real names and email addrs>
* Discussions-To:   <email address>
  Status:           <Draft | Active | Proposed | Deferred | Rejected |
                     Withdrawn | Final | Replaced | Obsolete>
  Type:             <Standards Track | Informational | Process>
* Requires:         <BIP number(s)>
* Replaces:         <BIP number>
* Superseded-By:    <BIP number>
```

Figure 1: Excerpt of BIP preamble structure from BIP2 [29].

which relies on an explicit model of variability in terms of features realized based on an integrated software platform [20, 28]. Recent literature also discusses more liberal approaches to managing software families, spanning a continuum that ranges from managing ad-hoc clone-and-own [35, 50, 53, 63] and feature toggling [41, 49] in distributed open-source communities to rigorous product-line engineering using a centrally managed integrated software platform [18, 52, 54]. Albeit at varying levels of systematic organization and pre-planning, it is the fundamental principle of reusing implementation artifacts that represents a common aspect across this continuum.

Interestingly, several vibrant ecosystems produce software families in a radically different way than classical variability-intensive systems. They are driven by factors other than software industrialization and mass customization, and exhibit variability that is not focused on reusing implementation artifacts. Instead, they focus on achieving interoperability within the software family through the ecosystem community's continuous effort to shape an open set of specification documents, referred to as *improvement proposals (IPs)*. Based on this set of IPs, developer groups within the



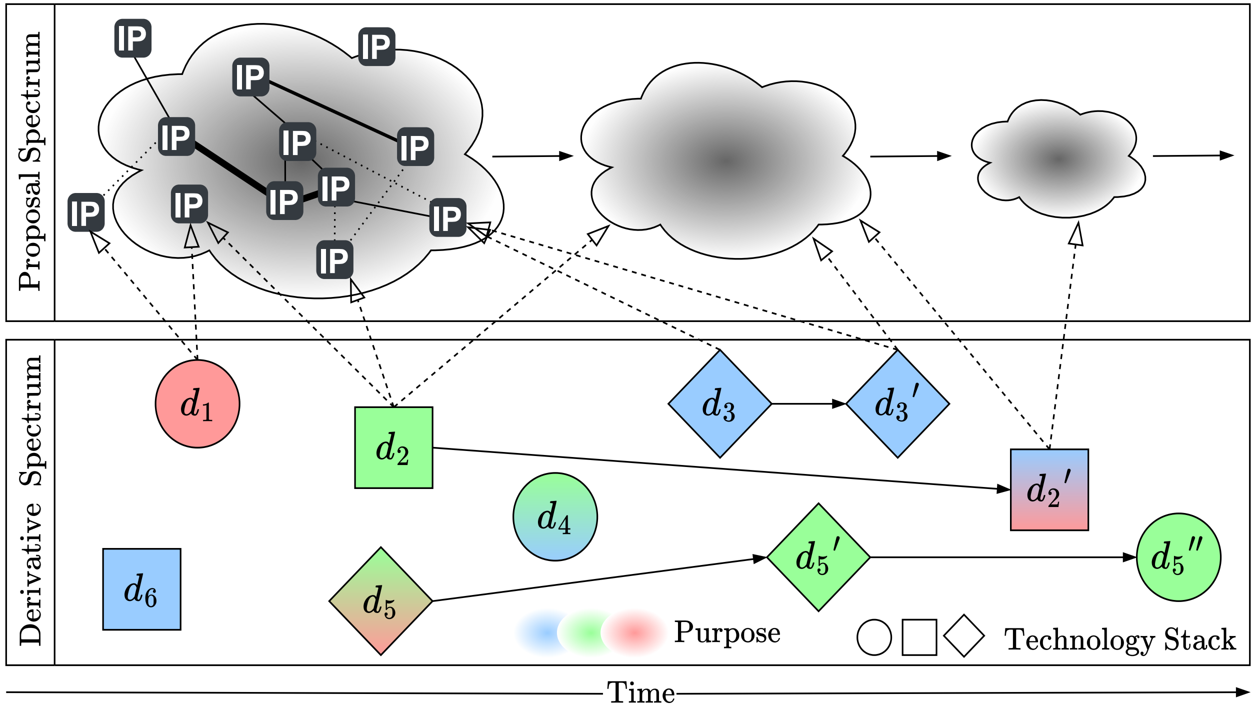
Mon 23 - Fri 27 June 2025 Trondheim, Norway

FSE 2025 (series) /

Ideas, **Visions** and Reflections

Community-Driven Variability

Preprint



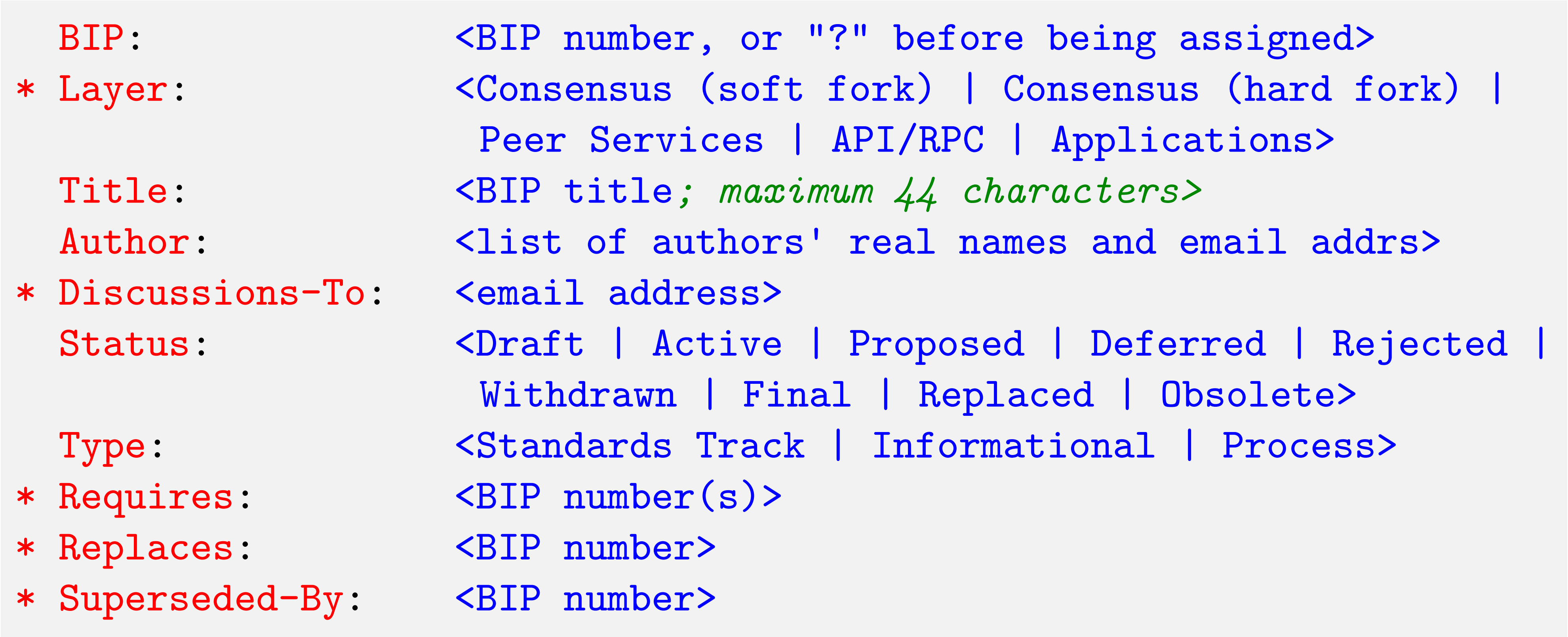
- **C1** Crowdsourcing
- **C2** Improvement Proposals
- **C3** Independent Derivatives
- **C4** Interoperability
- **C5** Decoupled Evolution

Boegli_2025_CDV_Preprint.pdf

- **P1** Missing overview in proposal spectrum
- **P2** Missing overview in derivative spectrum
- **P3** IP change impact assessment
- **P4** Misalignment of proposal/derivative spectrum
- **P5** Level of derivative interoperability
- **P6** Ecosystem forks

- **RG1** Systematic treatment of CDV in proposal spectrum
- **RG2** Supporting cohesive evolution of proposal and derivative spectrum
- **RG3** Methodical handling of derivative interoperability impairment

Appendix



```
  BIP:               <BIP number, or "?" before being assigned>
* Layer:             <Consensus (soft fork) | Consensus (hard fork) |
                      Peer Services | API/RPC | Applications>
  Title:             <BIP title; maximum 44 characters>
  Author:            <list of authors' real names and email addrs>
* Discussions-To:    <email address>
  Status:            <Draft | Active | Proposed | Deferred | Rejected |
                      Withdrawn | Final | Replaced | Obsolete>
  Type:              <Standards Track | Informational | Process>
* Requires:          <BIP number(s)>
* Replaces:          <BIP number>
* Superseded-By:     <BIP number>
```

Figure 1: Excerpt of BIP preamble structure from BIP2 [29].

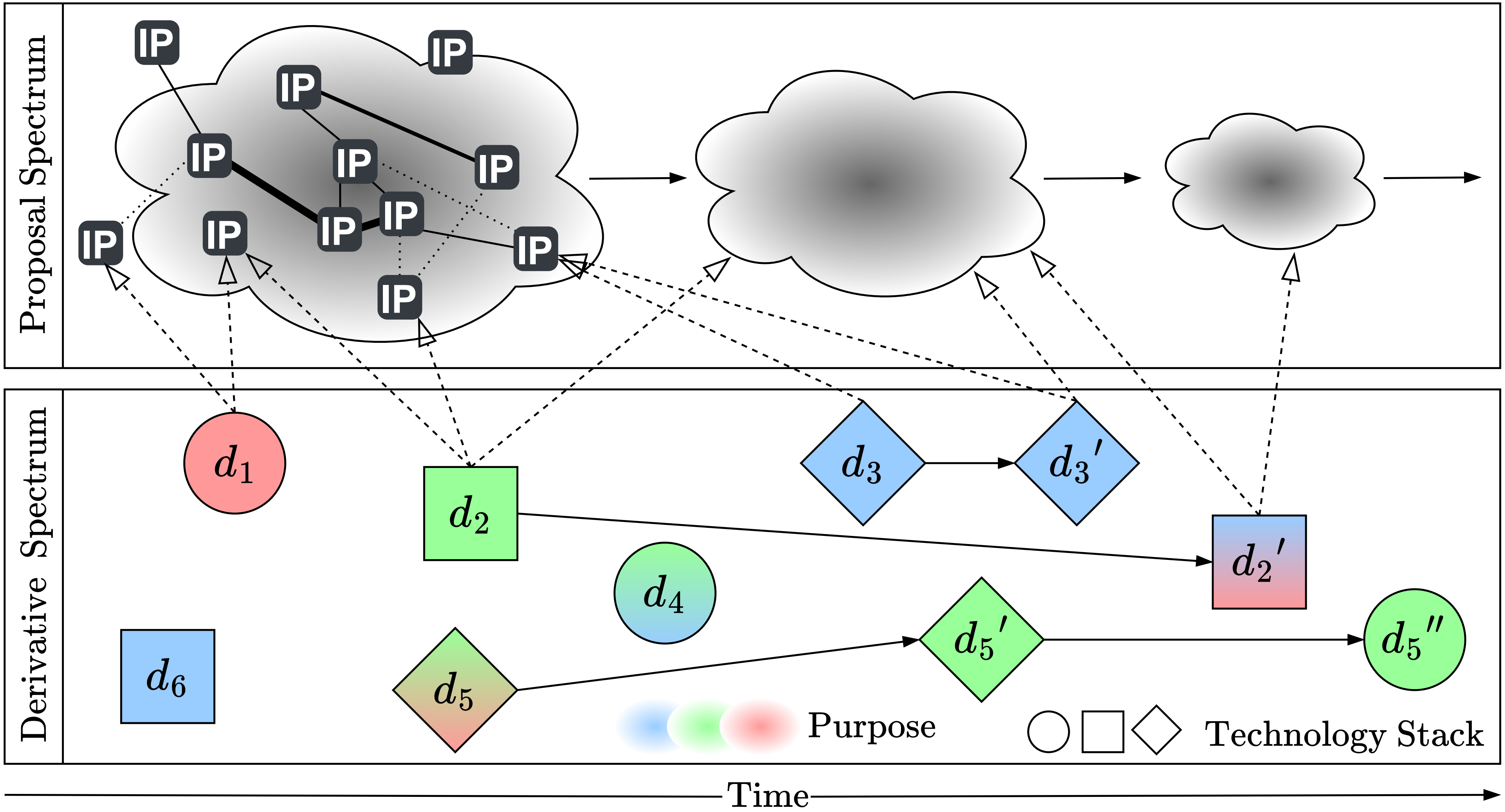
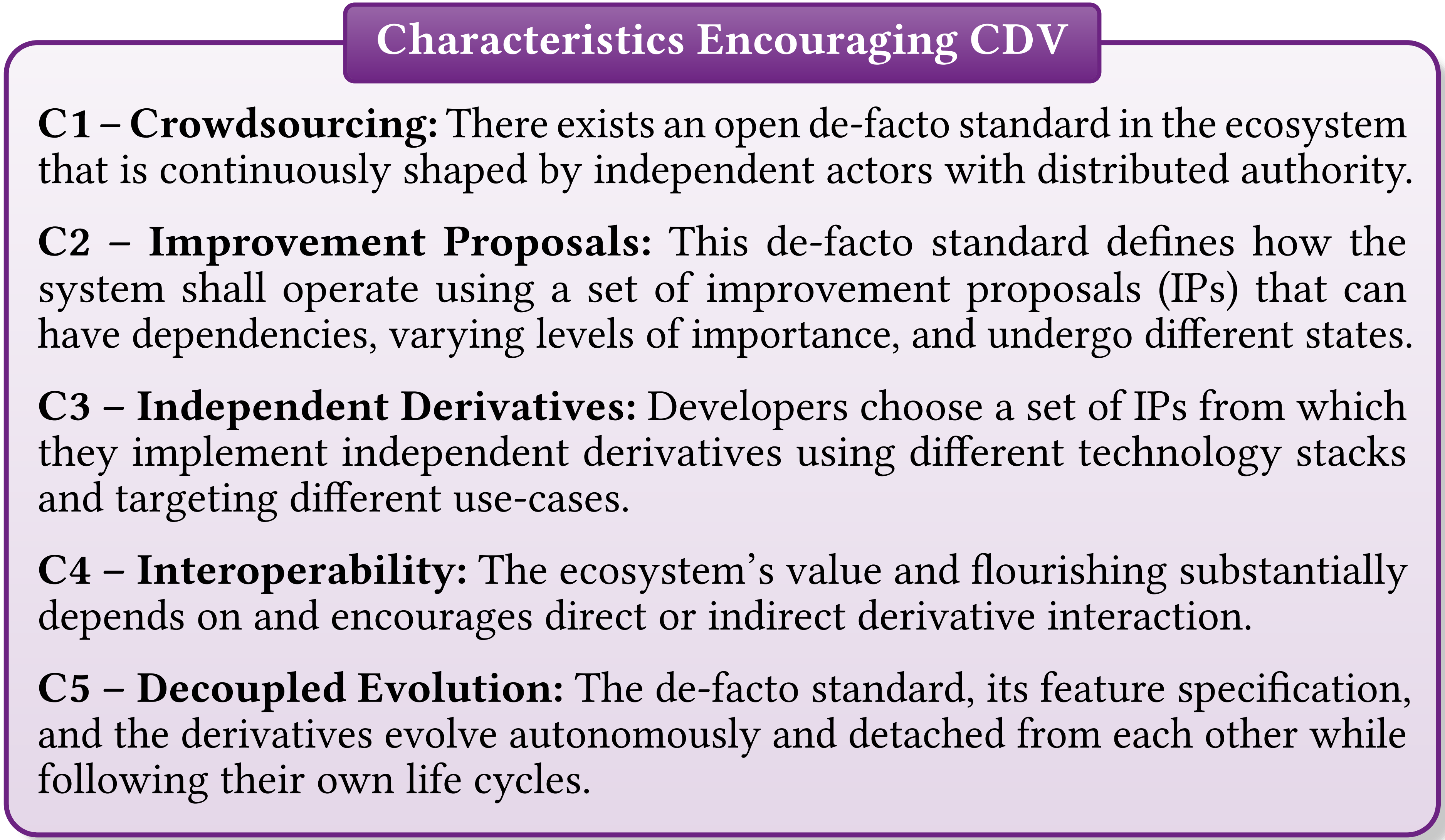


Figure 2: A schematic overview of the CDV landscape.



Characteristics Encouraging CDV

C1 – Crowdsourcing: There exists an open de-facto standard in the ecosystem that is continuously shaped by independent actors with distributed authority.

C2 – Improvement Proposals: This de-facto standard defines how the system shall operate using a set of improvement proposals (IPs) that can have dependencies, varying levels of importance, and undergo different states.

C3 – Independent Derivatives: Developers choose a set of IPs from which they implement independent derivatives using different technology stacks and targeting different use-cases.

C4 – Interoperability: The ecosystem's value and flourishing substantially depends on and encourages direct or indirect derivative interaction.

C5 – Decoupled Evolution: The de-facto standard, its feature specification, and the derivatives evolve autonomously and detached from each other while following their own life cycles.

Figure 3: Characteristics Encouraging CDV.

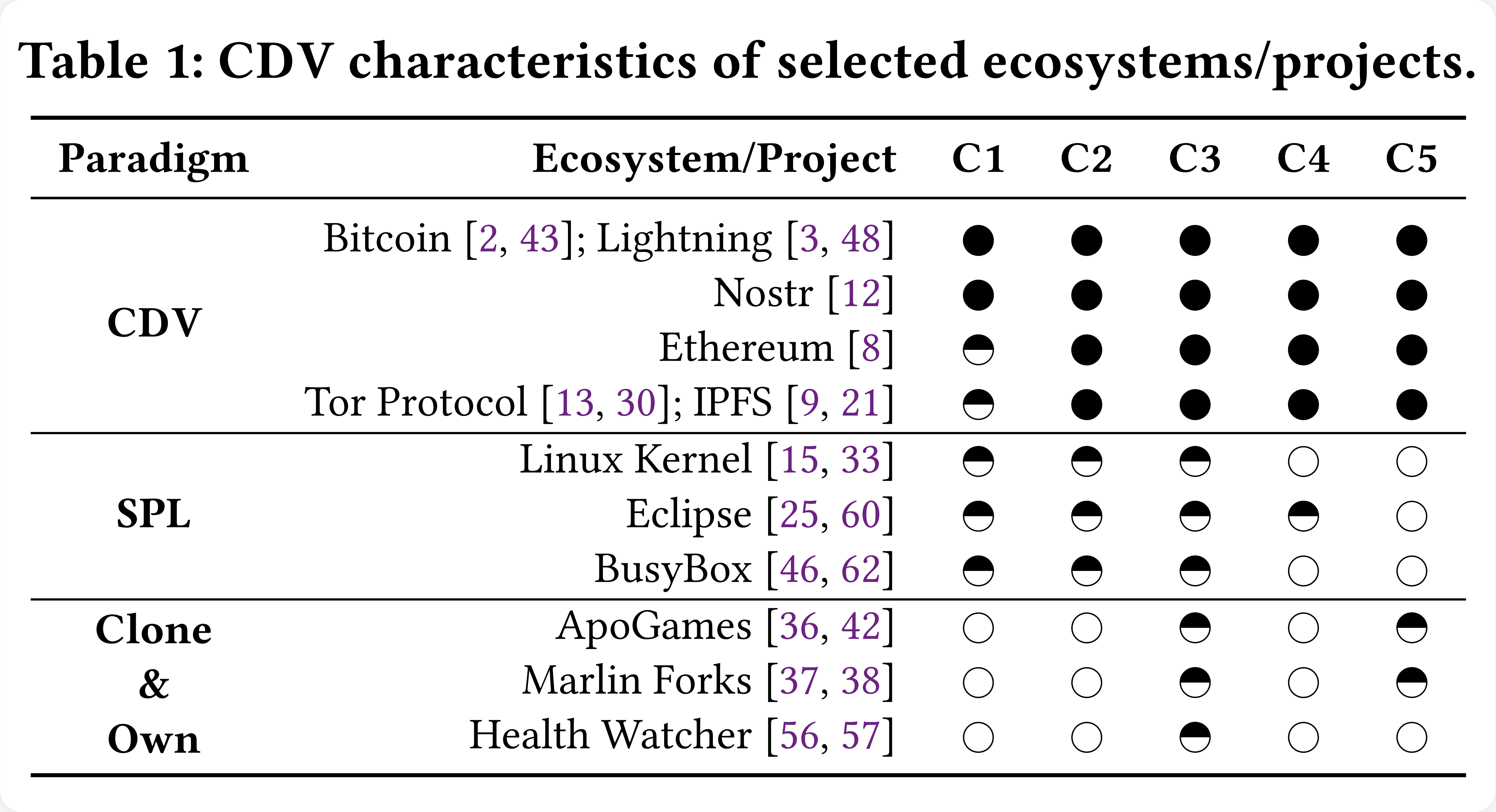


Table 1: CDV characteristics of selected ecosystems/projects.

Paradigm	Ecosystem/Project	C1	C2	C3	C4	C5
CDV	Bitcoin [2, 43]; Lightning [3, 48]	●	●	●	●	●
	Nostr [12]	●	●	●	●	●
	Ethereum [8]	◓	●	●	●	●
	Tor Protocol [13, 30]; IPFS [9, 21]	◓	●	●	●	●
SPL	Linux Kernel [15, 33]	◓	◓	◓	○	○
	Eclipse [25, 60]	◓	◓	◓	◓	○
	BusyBox [46, 62]	◓	◓	◓	○	○
Clone & Own	ApoGames [36, 42]	○	○	◓	○	◓
	Marlin Forks [37, 38]	○	○	◓	○	◓
	Health Watcher [56, 57]	○	○	◓	○	○

RG1 – Systematic treatment of CDV in proposal spectrum: Our first research goal is threefold. First, we aim to develop a variability modeling formalism and notation that can adequately capture CDV ecosystems and their evolution, providing a structured, explorable representation of the proposal spectrum amenable to analysis (P1). Second, we want to support the automated extraction of CDV models from various resources, with a focus on deriving variability models directly from IP collections. Third, analysis techniques shall be developed to reason about the structure and constraints of CDV models, spotting anomalous IPs and interrelations. This includes methods for differential analysis of CDV models representing different proposal spectrum snapshots, facilitating change impact analyses in the proposal spectrum (P3, P6).

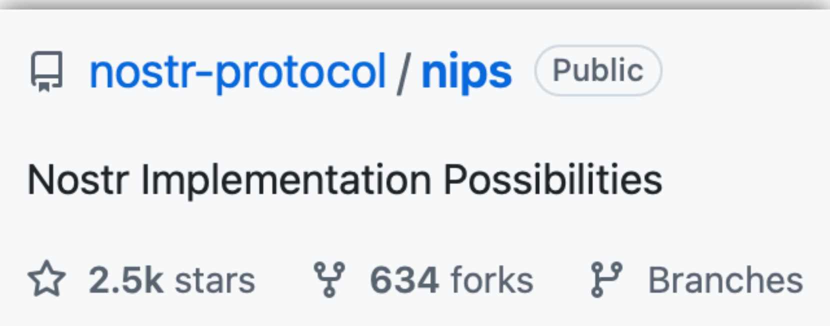
Impact: Holistic modeling of a CDV ecosystem's topology fostering comprehensibility and auditability.

RG2 – Supporting cohesive evolution of proposal and derivative spectrum: Given the autonomous evolution of these two spectra, our goal is to better understand and measure their cohesion (P4). This includes providing configuration support through CDV model-guided IP selection and first cohesion assessments by, e.g., checking a given set of IPs against a CDV model. However, the major endeavor pursued with this research goal is to support tracing of IPs from the proposal to the derivative spectrum, providing a better understanding of the derivative spectrum (P2) and facilitate further change impact analyses (P3). Besides IP traceability, we aim at mining CDV models from existing derivatives, enabling comparisons with those extracted from the IP spectrum (P4) and analyzing potential drift between community forks (P6).

Impact: Streamline the evolution of ecosystems by increasing the efficiency and effectiveness of future development endeavors.

RG3 – Methodical handling of derivative interoperability impairment: We dedicate our final research goal to address the challenges related to impaired interoperability within the derivative spectrum (P5), which boils down to handling and detecting undesired inter-derivative IP interactions. Anticipated interactions shall be documented and articulated through the CDV model, amenable to automatically validating derivatives wrt. proposal spectrum alignment (P4). Unanticipated interactions impairing interoperability shall be detected through systematic IP interaction testing, which must be both effective and efficient to be accepted in practice.

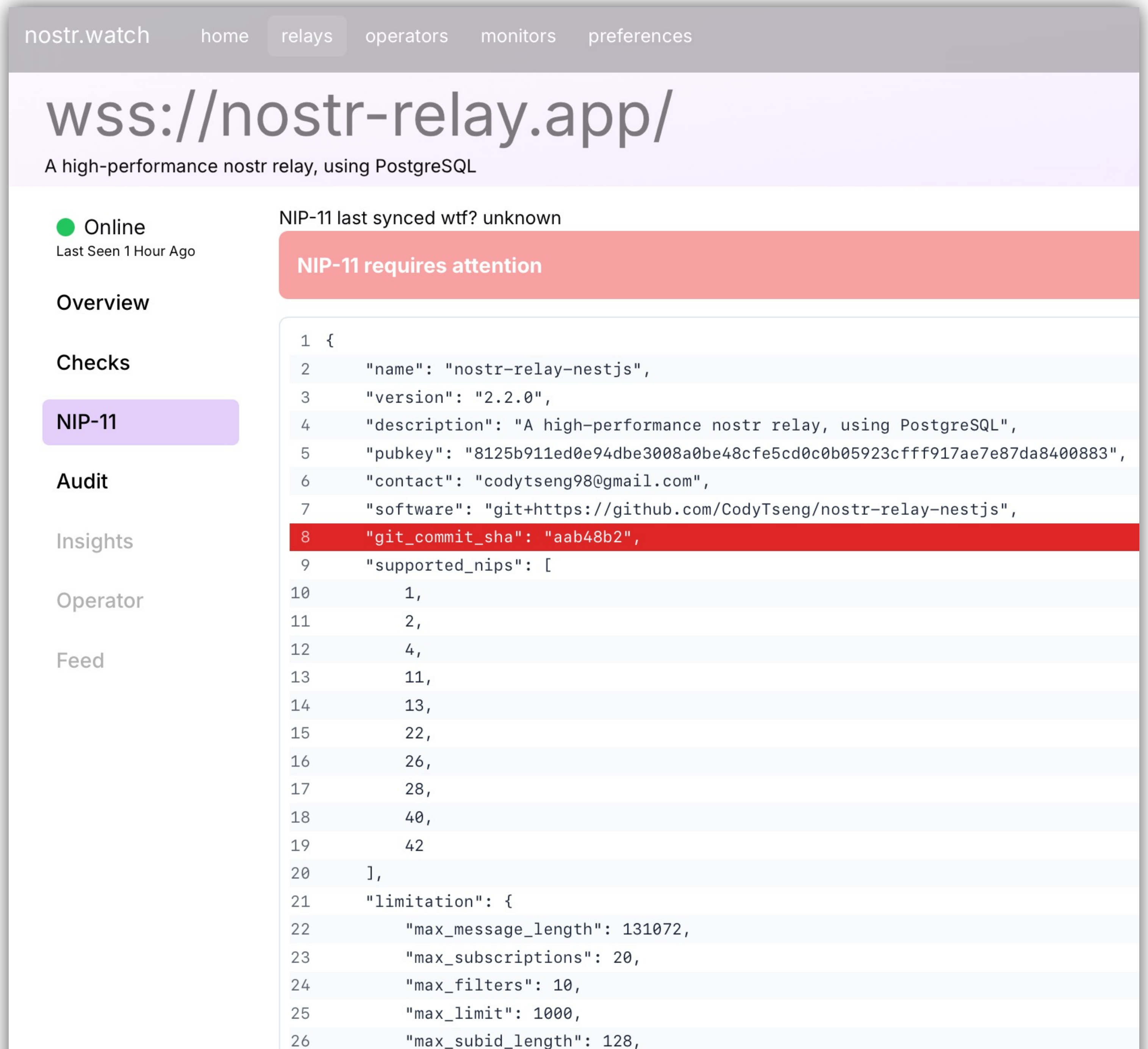
Impact: Reduce the effort and complexity of proper inter-derivative feature testing, further maximizing interoperability and positive user experience.



nostr-protocol / nips Public

Nostr Implementation Possibilities

2.5k stars 634 forks Branches



wss://nostr-relay.app/

A high-performance nostr relay, using PostgreSQL

● Online
Last Seen 1 Hour Ago

NIP-11 last synced wtf? unknown

NIP-11 requires attention

```
{
    "name": "nostr-relay-nestjs",
    "version": "2.2.0",
    "description": "A high-performance nostr relay, using PostgreSQL",
    "pubkey": "8125b911ed0e94dbe3008a0be48cfe5cd0c0b05923cfff917ae7e87da8400883",
    "contact": "codytseng98@gmail.com",
    "software": "git+https://github.com/CodyTseng/nostr-relay-nestjs",
    "git_commit_sha": "aab48b2",
    "supported_nips": [
        1,
        2,
        4,
        11,
        13,
        22,
        26,
        28,
        40,
        42
    ],
    "limitation": {
        "max_message_length": 131072,
        "max_subscriptions": 20,
        "max_filters": 10,
        "max_limit": 1000,
        "max_subid_length": 128,
```

Source: next.nostr.watch

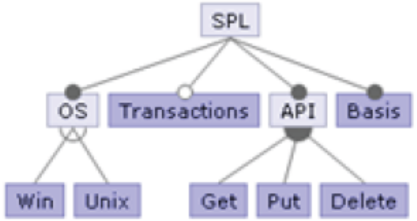
SPL

Problem Space

Solution Space

Domain Eng.

- Variability model (FMs)

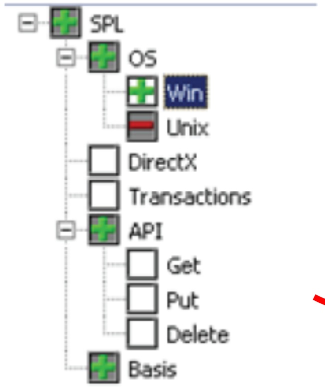


map

- Reusable implementation artifacts

App. Eng.

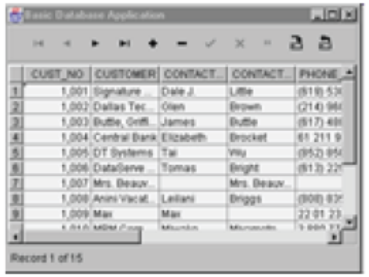
- Configuration (feature selection)



- ☑ SPL
  - ☑ OS
    - ☑ Win
    - ☒ Unix
  - □ DirectX
  - □ Transactions
  - ☑ API
    - □ Get
    - □ Put
    - □ Delete
  - ☑ Basis

- Generated product

generate



Basic Database Application

	CUST_NO	CUSTOMER	CONTACT...	CONTACT...	PHONE_...
1	1,001	Signature ...	Dale J.	Little	(619) 53
2	1,002	Dallas Tec...	Glen	Brown	(214) 96
3	1,003	Buttle, Griffi...	James	Buttle	(617) 48
4	1,004	Central Bank	Elizabeth	Brocket	61 211 9
5	1,005	DT Systems	Tai	Wu	(852) 85
6	1,006	DataServe ...	Tomas	Bright	(613) 22
7	1,007	Mrs. Beauv...		Mrs. Beauv...	
8	1,008	Anini Vacat...	Leilani	Briggs	(808) 83
9	1,009	Max	Max		22 01 23
	1,010	MPM Corp...	Miyuki	Miyamoto	3 880 77

Record 1 of 15

CDV

Problem Space

Solution Space

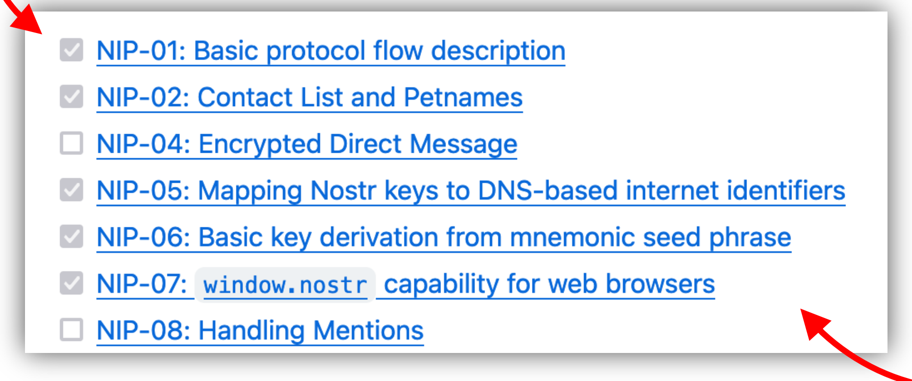
Domain Eng.

- Collection of specification documents / improvement proposals / implementation possibilities
- BIPs, NIPs, EIPs, etc.

???

select

App. Eng.



☑ NIP-01: Basic protocol flow description

☑ NIP-02: Contact List and Petnames

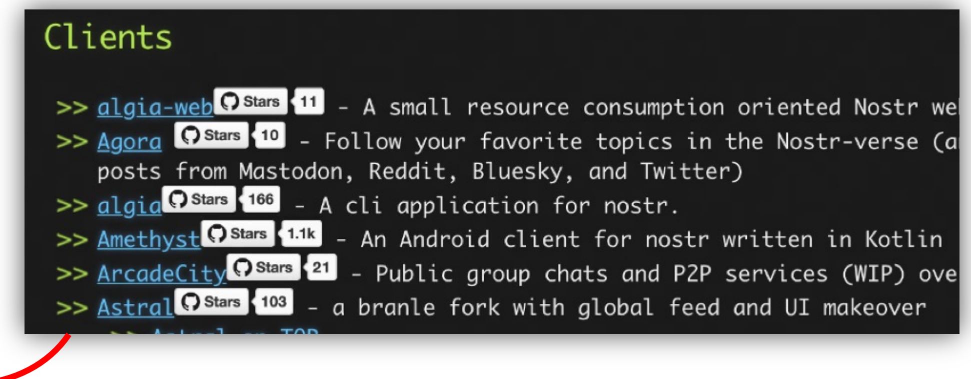
□ NIP-04: Encrypted Direct Message

☑ NIP-05: Mapping Nostr keys to DNS-based internet identifiers

☑ NIP-06: Basic key derivation from mnemonic seed phrase

☑ NIP-07: `window.nostr` capability for web browsers

□ NIP-08: Handling Mentions



Clients

```
>> algia-web Stars 11 - A small resource consumption oriented Nostr we
>> Agora Stars 10 - Follow your favorite topics in the Nostr-verse (a
   posts from Mastodon, Reddit, Bluesky, and Twitter)
>> algia Stars 166 - A cli application for nostr.
>> Amethyst Stars 1.1k - An Android client for nostr written in Kotlin
>> ArcadeCity Stars 21 - Public group chats and P2P services (WIP) ove
>> Astral Stars 103 - a branle fork with global feed and UI makeover
```

implement

Thanks

b

UNIVERSITY
OF BERN