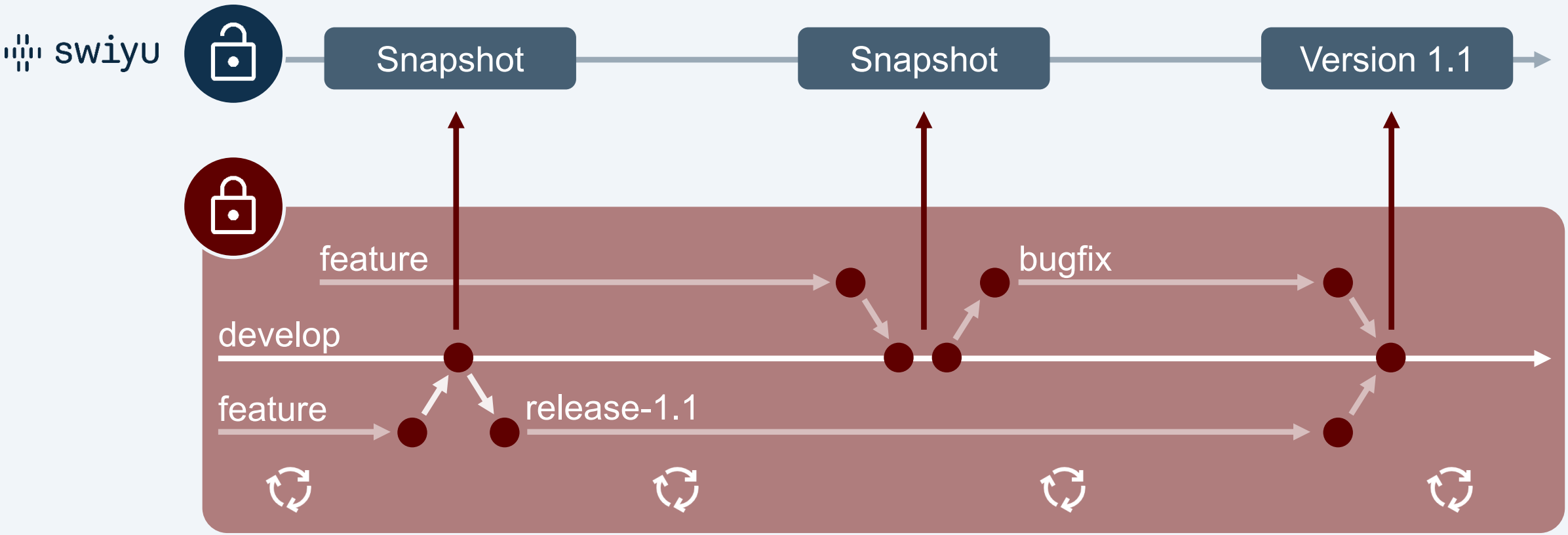
Neues auf GitHub

Entwicklungsprozess & Open Source

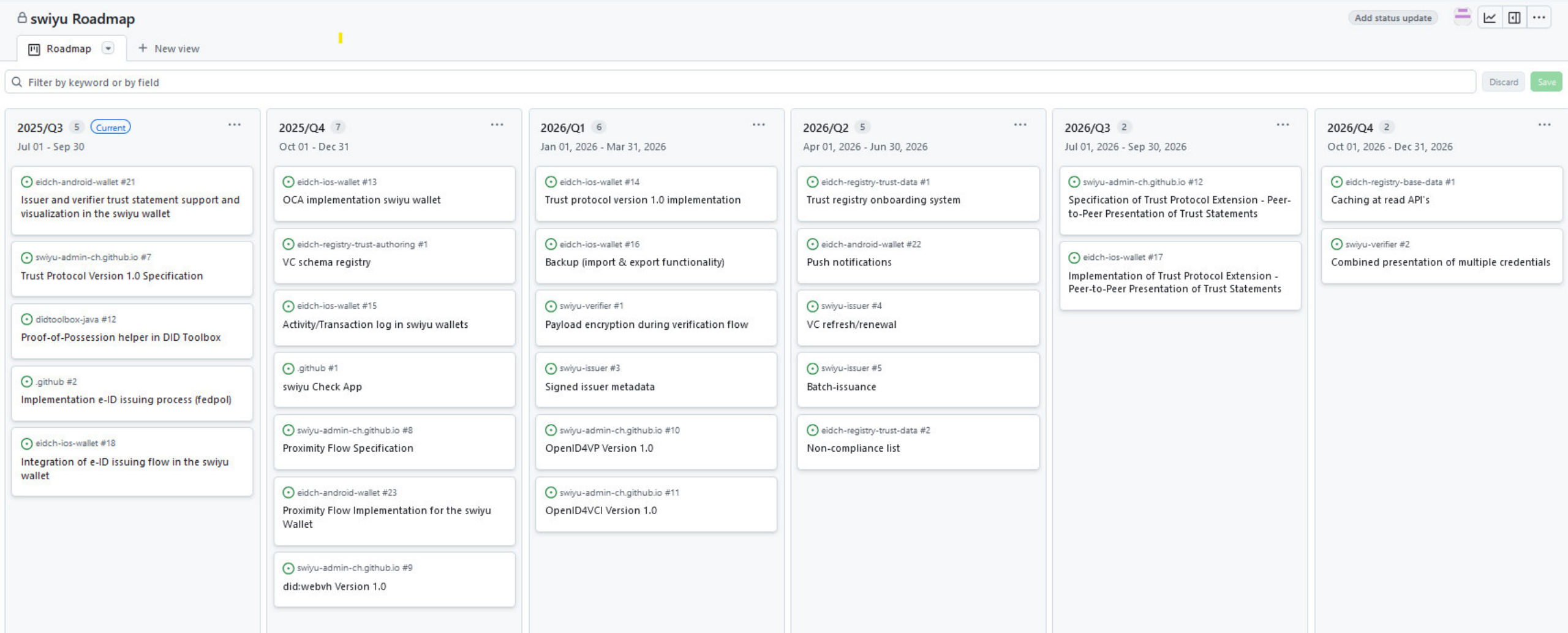


Neue GitHub Organisation für Public Beta

github.com/swiyu-admin-ch	• Übersicht und generelle Einführung • «Discussions» • Neu: «Projects»
/community	• Folien des Partizipationsmeetings • Technische Konzepte
/swiyu-admin-ch.github.io	• Technische Dokumentation & Standards • Cookbooks
Code repositories	• Quellcode der Vertrauensinfrastruktur • «Issues» für Probleme und Wünsche

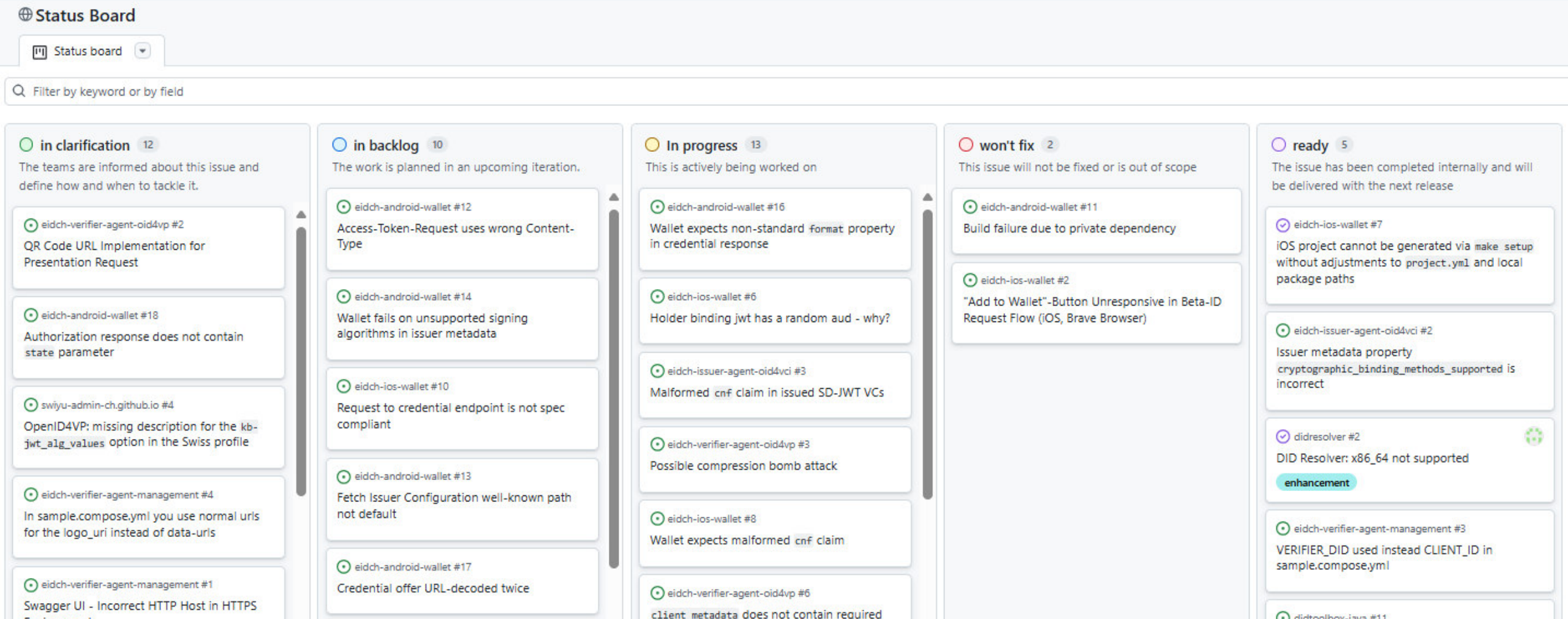
GitHub Project für die Feature Roadmap

Über swiyu-admin-ch, Tab «Projects» (https://github.com/orgs/swiyu-admin-ch/projects/1/views/7)



GitHub Project für das Status Board

Über swiyu-admin-ch, Tab «Projects» (https://github.com/orgs/swiyu-admin-ch/projects/2/views/2)



«Known Issues» für Bug Bounty Programm

- Erste uns bekannte Fehler sind bereits mit dem Issue Type «KnownIssue» erfasst

- Für das Bug Bounty Programm wurden weitere Issues veröffentlicht
  - Security Findings aus Pentests mit externer Firma
  - Identifizierte potenzielle Schwachstellen aus Masterarbeit

- Weitere Erkenntnisse aus dem Bug Bounty Programm können folgen

Release Ankündigungen & Betriebsstatus

Neue Kanäle für Ankündigung kommender Releases

- Diskussions-Thread auf GitHub: https://github.com/orgs/swiyu-admin-ch/discussions/11
  - «Subscribe» mit GitHub Account -> Information per E-Mail

- Fusszeile der swiyu technical documentation https://swiyu-admin-ch.github.io/release-announcements/
  - RSS-Feed abonnieren

Informationskanal für allfällige Betriebsunterbrüche

- Diskussions-Thread auf GitHub: https://github.com/orgs/swiyu-admin-ch/discussions/12
  - «Subscribe» mit GitHub Account -> Information per E-Mail

Ausblick: Neue Issuer- & Verifier-Repositories

- Die Repositories für den Management- und den Signer-Service werden zusammengefasst
- Das gleiche gilt für den Management- und Validator-Service
- Gründe sind u.a.
  - Vereinfachung der Codebasis
  - Beseitigung von Redundanzen
  - Höhere Stabilität
- Die neuen Repositories sollten ca. Mitte August veröffentlicht werden
- Gemeldete Issues werden mit dem Release der neuen Repositories behoben
- Die Cookbooks werden entsprechend angepasst

Upgrades & Breaking Changes

Problematik von Änderungen und Upgrades

- Upgrades und Änderungen zum Teil unvorhersehbar/unbeeinflussbar, insbesondere bei internationalen Spezifikationen
- Änderungen (Breaking Changes) können Interoperabilität von Komponenten verhindern
- Ständige Evolution lässt Breaking Changes kaum verhindern
- Falscher Umgang mit Breaking Changes kann zu Downtimes führen

➡ **ZIEL**
Implementierung von Breaking Changes, auf eine **non-Breaking** Weise

➡ **LÖSUNG**
Expand-Migrate-Contract Architekturmodell, um Änderungen schrittweise und abwärtskompatibel umzusetzen und Disruptionen zu vermeiden.

Expand-Migrate-Contract Schema für einen reibungslosen Change

Expand

- Hinzufügen neuer Elemente (z.B. Funktionalitäten/ Komponenten/Spezifikationen)
- Aufbau von Abwärts-kompatibilität, sodass alte Funktionalitäten/Komponenten parallel bestehen können

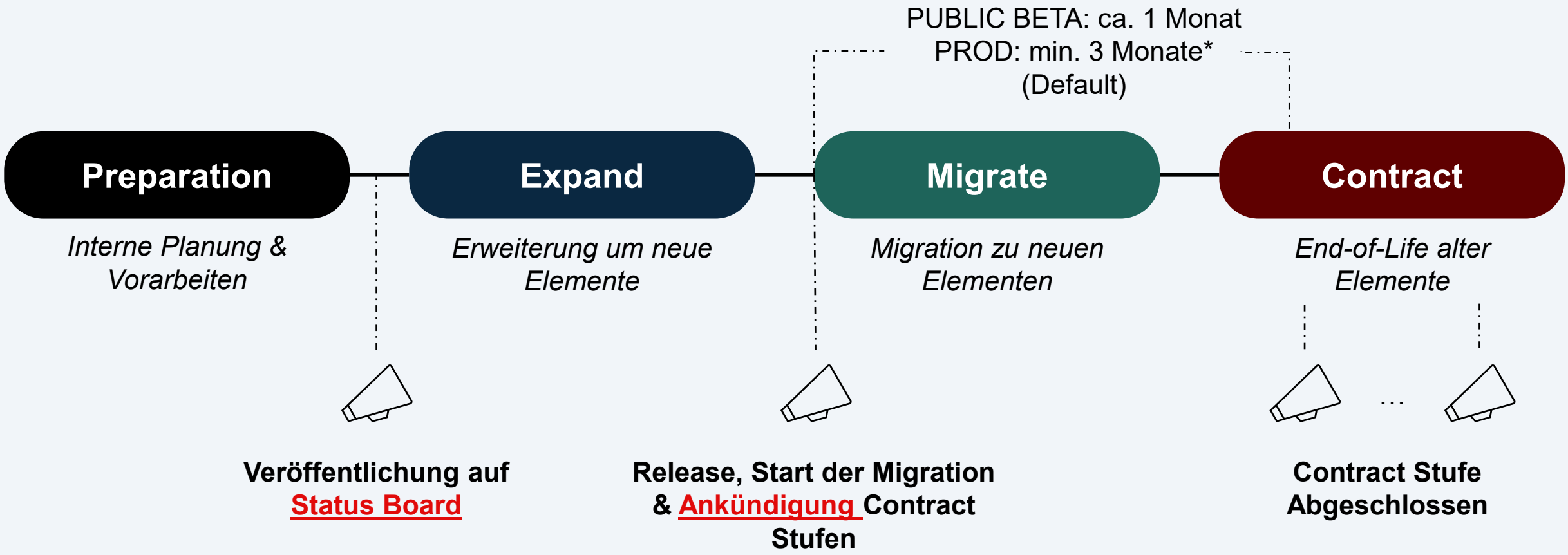
Migrate

- Migration von den alten zu den neuen Elementen
- Bevorzugte Nutzung der neuen Elemente
- Abwärtskompatibilität bleibt über diese Phase hinweg erhalten

Contract

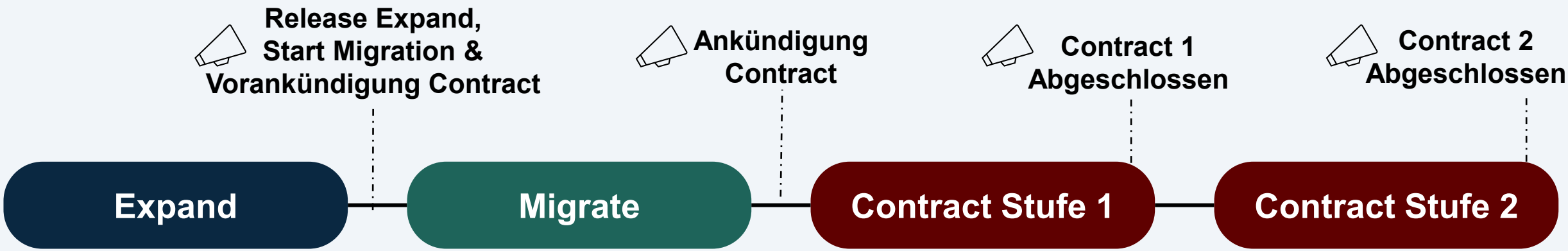
- Stufenweises Einstellen der Abwärtskompatibilität
- Nicht immer für alle Changes und Komponenten vorhanden

Zeitliche Abfolge und Kommunikation



*Zeitplan für Backend Komponenten;
Kürzere Zyklen bei swyiu Wallet durch Version Enforcement möglich

Beispiel– Fix: Wallet unterstützt falschen `cnf` Claim



Wallet:

Richtige und falsche `cnf` Claims werden unterstützt

Generic Verifier:
Beide `cnf` Claims werden unterstützt

Wallet:

Enforcement der Expanded Version, d.h. beide `cnf` Claims werden unterstützt

Generic Verifier:
Adaption des Updates in der Community

Generic Issuer:

nur noch richtige `cnf` Claims

Adaption des Updates in der Community sicherstellen

Wallet (Android & iOS):

nur noch richtige `cnf` Claims

Generic Verifier:

nur noch richtige `cnf` Claims

Adaption des Updates in der Community sicherstellen

Strategie und Umgang mit Spezifikationen

Strategie für und nach Go Live 2026

Für Go Live 2026

- **Mindestens Version 1.0** bei allen Spezifikationen
- V1.0 jedoch noch nicht bei allen Spezifikationen vorhanden!
- Sobald V1.0 bekannt, **zeitnahes Upgrade** der jeweiligen Spezifikation

Regelmässige Upgrades nach Go Live

- Möglichst **zeitnahe** Unterstützung **neuer Versionen**
- Aber **kein «blindes» Updaten**, ohne Auswirkungen auf swiyu zu evaluieren
- **Abwärtskompatibilität** und **EMC-Pattern** sollen Interoperabilität gewährleisten

Aktueller Stand der Spezifikationen

Spezifikation	Aktuell	Public Beta	
DID Core	v1.0	v1.0	✓
DID:webvh	v1.0	v0.3	Expand-Phase: Version 1.0
OpenID4VCI	draft 15	draft 13	Version 1.0 erwartet in Q3; Aktuell Vorbereitungen für Batch Issuance
OpenID4VP	draft 29	draft 20	Version 1.0 erwartet in Q3
OCA	v2.0	v1.0	Eigene Extension von OCA v1.0; Implementierung läuft
Swiss Trust Protocol	v0.1	v0.1	Version 1.0 in Erarbeitung
SD-JWT VC	draft 10	draft 4	Wenig Indikationen zu Version 1.0
SD-JWT	draft 22	draft 10	Hohe Dynamik in der Spezifikation; Wenig Indikationen zu Version 1.0
Token Status List	draft 11	draft 3	Wenig Indikationen zu Version 1.0

Upgrade auf DID:webvh v1.0

Expand

Umfang

- Vollständiges Upgrade von 0.3 auf 1.0 der vom Bund bereitgestellten Komponenten
- Inkl. eingesetzter optionaler Features
- Keine neuen optionalen Features
- Sicherstellen der Abwärtskompatibilität

Komponenten

- Basisregister
- DIDToolbox/-Resolver

Migrate

Umfang

- Abwärtskompatibilität vorhanden
- Upgrade/ Migration der von Ökosystem Teilnehmern eingesetzten Komponenten
- Alte DIDs sollen, wo möglich, durch neue DIDs abgelöst werden

Komponenten

- Issuer
- Verifier
- Wallet

Contract

D.h.

- Abwärtskomp. bleibt vorerst erhalten
- DIDs bleiben vorerst gültig.
- eLFAs bleiben gültig.

Komponenten

- Basisregister
- DIDToolbox/-Resolver
- Issuer/Verifier/Wallet

Feedback und Fragen