mojaloop

Upgrading Mojaloop instances

Issues, strategy and policy

What’s the problem?

Pretty soon, people in entire national communities are going to be relying on Mojaloop systems to support their business and personal lives.

So, a Mojaloop system must be available 24*7*365

What's the problem?

Pretty soon, people in entire national communities are going to be relying on Mojaloop systems to support their business and personal lives.

So, a Mojaloop system must be available 24*7*365

The community will be continuing to develop major functional improvements to the OSS code

Individual implementations will need to work to incorporate these improvements in their implemented code

What’s the problem?

Pretty soon, people in entire national communities are going to be relying on Mojaloop systems to support their business and personal lives.

The community will be continuing to develop major functional improvements to the OSS code

Individual implementations will need to work to incorporate these improvements in their implemented code

So, a Mojaloop system must be available 24*7*365

So the problem is: what actions can (or should) this community take to support the deployment of *breaking changes* in systems *which can’t be taken down*?

Breaking changes

- Changing versions of the switch
- Changing requirements on participants

Systems that can’t be taken down

- Problems with persistence
- Problems with synchronisation

What kinds of things could the community do?

- Reduce the impact of functional change

What kinds of things could the community do?

- Reduce the impact of functional change
- Insist on the deployability of technical deliverables (what does "*done*" look like?)

What kinds of things could the community do?

- Reduce the impact of functional change
- Insist on the deployability of technical deliverables (what does “*done*” look like?)
- Become a world leader in low-impact deployment techniques (as measured by…?)

Versioning the Mojaloop product

How should we version Mojaloop releases?

New functionality which might require versioning:

- Settlement v2
- Ongoing FX changes
- Cross currency / network
- Updated features, enhancements
- Newer releases of the APIs – FSPIOP, Settlement, Operations / Admin

What strategies might we adopt?

- Minor version change guideline
  - Optional input parameters such as query strings added in a request
  - Optional parameters added in a request or a callback
  - Error codes added
- Major version change guideline
  - Mandatory parameters removed or added to a request or callback
  - Optional parameters changed to mandatory in a request or callback
  - Parameters renamed
  - Data types changed
  - Business logic of API resource or connected services changed
  - API resource/service URIs changed

Mojaloop Versioning: Considerations

1. Standardize branching, tagging mechanism on GitHub
2. API versioning standards
3. Version negotiation in the API - to be added
4. Handling major, minor versions in the implementation – Guidelines for schemes
5. Supported versions in implementation - latest versions
6. Timelines for support

The floor is yours…