Web Zero, with **m-ld**[1]

project delta

Hypothesis

The first implementation of the Web was read-only, and while content editing has become a feature since ("Web 2.0"), it has been fenced into isolated 'platforms'. We will demonstrate a secure re-democratised version of the read/write Web, with multi-collaborator editable apps and data that can be owned by anyone, backed by universal identifiers, a standard data representation, and access control; in which no page or dataset is a silo. We call this "Web Zero".

In this vision, data is stored and used in *reactive*, *replicated Linked Data* datasets. By *Linked Data*, we mean named with universal identifiers, using HTTP URLs as much as possible, in RDF-compatible, machine-readable formats[2]. By *replicated*, we mean that applications working with the data are able to create a local, read-writeable copy which remains eventually consistent with other copies, with low latency (when the network allows). By *reactive*, we mean that each application is able to immediately react to and reflect changes in that local copy, whether caused by local operations or operations in other copies.

Together, these properties support open collaboration across the Web. For instance, in Linked Data Platform[3], an RDF Source is a resource which can be fully represented as RDF data. LDP specifies means for updating such a resource using PUT and PATCH methods, but offers no means to collaborate in real time on editing such a resource. We intend to demonstrate that such a specification would be both possible and useful.

We will not produce a complete specification in this project, as that work should be done by a broader working group. Instead, we will produce a set of concrete software components which demonstrate that such an approach is practical, and which will make it simple for developers to create such collaborative applications over Linked Data resources (such as Solid and Linked Data Platform Resources) with compelling, responsive user interfaces. These libraries will not be mere proofs of concept — we expect developers to use these libraries in actual applications.

Delta from 2022 draft

The primary focus of the project remains to deliver content editing as a first-class property of the Web.

In our analysis, we have noted that Web-delivered software today is frequently oriented to 'apps', rather than 'pages'. The greatest value we can offer to the Web with our core

---

[1] NLnet ref. 2022-02-056
[2] https://www.w3.org/DesignIssues/LinkedData.html
[3] https://www.w3.org/TR/ldp/

technology, **m-ld**, is a developer-friendly solution for highly-interactive local-first Web applications using Linked Data semantics.

> Therefore, our choice is to focus on closing the gap between our core capability and the most impactful modern tooling for Web apps. We will create a layered open-source library, starting with core **m-ld**, building on that with support for ReactiveX Observables, and then providing React.js hooks to use those Observables – a layer that could be easily reimplemented for any framework.The two different apps will be able to share live data

Local-first web applications can still benefit from secure durable storage of data on a server. They also rely on a message delivery mechanism, which in practice must be either a deployed service (like a message broker) or a cloud service. While these are cheap to set up, they may become expensive in the long run due to operational costs, provider markup and service over-specification.

We will provide and run an open-source cloud service that fulfils these requirements for collaborative web apps using **m-ld**, to eliminate getting-started overhead for trials, research, personal projects and startups, and to ultimately provide a basis for revenue from scale-ups and enterprises. In our demonstration we will emphasise that the Gateway is used for convenience and data durability, but is optional, and could be self-hosted.

We will still demonstrate a secure collaborative editable Web – our demo will build an 'app' that uses its own code, via React, to manipulate the DOM.[4] This app will not require a 'backend' data store, but a clear direction will be provided for apps that wish to use one.

2022 proposal	**2023 project plan changes**
Functional specification - 5 days - € 2.600	Now together in the **Analysis** milestone
Project establishment and outreach - 5 days - € 2.600	
DOM ↔ RDF translation (code + unit tests) - 15 days - € 7.800	We will not address DOM manipulation directly, as it is better done using existing frameworks & libraries. We will demonstrate example UI elements, including for conflict resolution.
Page editing controls (minimal) - 10 days - € 5.200	
Integration with OIDC provider - 10 days - €5.200	In this project we will develop examples and tests to show identity integration, in **Security Support**. Specific UI controls will be the choice of the app.
Security controls ("sharing") - 5 days - €2.600	
System tests 10 € 5.200	Suitable testing is assumed for all code components.

[4] More detail on the motivation for this can be found here.

2022 proposal	2023 project plan changes
SHACL (subset) schema constraints 15 €7.800	Replaced with the more general **Data Types Support**. Static types are more important to our audience than runtime dynamic types.
Binary literals 10 € 5.200	Approach to be considered in the **Text Editing Support** milestone (which has similar needs), but not to be engineered in full.
Text CRDT integration (timeboxed) 10 €5.200	This remains a part of **Text Editing Support**, which also includes library support.
- new for 2023 -	Added **Collaborative Web Library** (see below)
- new for 2023 -	Added **Gateway** (see below)

Planning

See accompanying project plan document.