

Integration of open decision-making data and mapping module

Decentralised Citizens Engagement Technologies
Specific Targeted Research Project Collective Awareness Platforms



Creative Commons
Attribution-NonCommercial-
ShareAlike 4.0 International
License

The logo for 'd-cent' features a lowercase 'd' inside a black circle, followed by the text '-cent' in a bold, sans-serif font.

d-cent

FP7 – CAPS
Project no. 610349
D-CENT
Decentralised Citizens
Engagement Technologies

Lead beneficiary:
Open Knowledge Foundation

D5.2
Integration of open
decision-making data
and mapping module
Sept 2015
Version Number: 1

Authors:

Jaakko Korhonen (OKFFI)
Linda Roy (ThoughtWorks)
John Cowie (ThoughtWorks)
Sander van der Wall (OKF)

Editors and reviewers:

Robert Bjarnason (Citizens Foundation)
Harry Halpin (ERCIM)
Francesca Bria (Nesta)

The work leading to this publication has received funding from the European Union's Seventh Framework Programme (FP7/2007 – 2013) under grant agreement n° 610349.

The content of this report reflects only the author's view and that the Commission is not responsible for any use that may be made of the information it contains.



Open Knowledge
Foundation



Contents

1. Executive summary	3
1.1. High Level Feature Overview	3
1.2. Open Data	4
1.3. Crowdsourcing.....	4
1.4. Map.....	5
2. Integration with other tools.....	6
2.1. ActivityStream integration with Maps and Stonecutter	6
2.2. Crowdsourcing, integration of ActivityStream and Input from Decision articles	7
2.3. Crowdsourcing, integration of notifications and comments.....	7
3. Integration Design	8
4. User Testing: Finnish Pilot.....	10
4.1. Pilot progression this far	10
4.1.1. The Open Ahjo API.....	12
4.1.2. The Current Version	12
4.1.3. The Next Steps	13
4.2. ActivityStreams aggregation pilot.....	14
4.3. Local refugee co-op pilot.....	14
5. Technology Rationale	15
6. Security	16
7. Deployment	17
8. Database Design	18

1. Executive summary

This deliverable involves integrating the different D-CENT components based on the experience in previous D-CENT work in WP5 (D5.1; D5.3; D5.4).

As W3C Social Web working group input with ActivityStreams documented in D5.1 (<http://dcentproject.eu/wp-content/uploads/2015/02/D5.1-Pilot-Implementation-of-Open-Social-Web-for-Participatory-Democracy.pdf>) appears to be concretely solving more problems than expected, we are focusing the efforts in realising these possibilities in an integrated solution within D-CENT, rather than maintaining authentication and data connection integrations to external tools.

The core of D5.2 is publishing Open Data, and the integration of crowd-sourcing functionalities into D-CENT. It has turned out that ActivityStream 2 API can be used directly as an Open Data Store, so further applications like CKAN or Pybossa can directly connect to that. In context of Decision Data Crowdsourcing the needs will be fulfilled by providing the possibility to feedback comments and metadata to the Activitystream, thereby enriching the open data being published. The same data will be published as comments to an article, and visualised on a map page.

The concrete use case needs and opportunities for commenting, crowdsourcing keywords and sharing items have risen through the experiences of piloting Decisions interface in Helsinki as output from D5.1.

Technical implementation of the features is still on varying levels, mostly pending work in the secure notification provider (D5.6) and will be implemented synchronously with that development. Moreover, a full AS2 vocabulary is expected to be released in October, which might still have some impact on the piloting.

1.1. High Level Feature Overview

We are using the ActivityStream 2.0 machine readable web service to integrate and aggregate data in different contexts to create a multi-channel multi-organisation participation experience. Through reusing the components created in previous deliverables in the project, we are creating an integrated experience where parties contribute to the cyclic creation, use, reuse, and enriching of open data.

There are four key features introduced:

1. Open Data datastore and application interface (API). This is used to aggregate social media data in between D-CENT nodes and 3rd party organisations' systems that are complying with the W3C ActivityStreams 2.0 (AS2) standard.
2. Crowdsourcing functionality to reuse and enrich the social media data to improve its quality.
3. Map visualisation to print items in the ActivityStream on a map, to enable local real-life interaction in between users.
4. A limited set of API data connectors, developed in D5.1, are to be tested with Stonecutter (D5.4) and Mooncake (D5.6) applications, to create an integrated experience.

1.2. Open Data

The core of D5.2 is the integration of crowd-sourcing Open Data into D-CENT via the deep integration of open datastore. It has turned out that an ActivityStream 2 API can be used directly as Open Data Store, so further applications like CKAN can directly connect to that.

Feature definition: Publishing Open Data from the ActivityStreams server API in compliance with the Open definition with <http://opendefinition.org/>

User Story: As a member of Developer Community, I need access to contents to create my own applications.

Description: Each of the D-CENT modules come with a AS2 API, so aggregated data from other nodes can be published in an organisations own D-CENT node. Streams from several cities or even a national or European-wide social media newsfeed can be aggregated, and views to that can be created topically in whichever of the D-CENT nodes.

Technical implementation: The decision-making data activity stream ActivityStreams 2.0 (AS2) compliant application interface (API) through which data can be accessed, and aggregated for reuse. For each D-CENT node, there is a machine-readable AS2 compliant API to enable access to the data. A proof-of-concept was published as part of D5.1: <https://github.com/samuellmr/activitystore-poc>

1.3. Crowdsourcing

The second deliverable the integration of crowdsourcing comments as Open Data into D-CENT ActivityStream.

Feature definition: Crowdsourcing of document metadata through commenting and hashtagging.

User Story: As a commenter, I want my comments and hashtags to be reused for publishing the documents I have commented, to other interested users.

Description: Data from user interaction will be written back to the ActivityStream for reuse. Comments and hashtags will be used to enrich the data, and improve further search results' semantic match to user interest. Co-created articles can be stored in the ActivityStream as alternate versions of the original. The functionality enables further data crowdsourcing user stories to be developed with a similar data reuse cycle.

data and mapping module

Technical implementation: Data is read from ActivityStream datastore, and user input is written back to the ActivityStream datastore, linked to the original data. A restful Javascript is used to access the AS2 JSON API.

1.4. Map

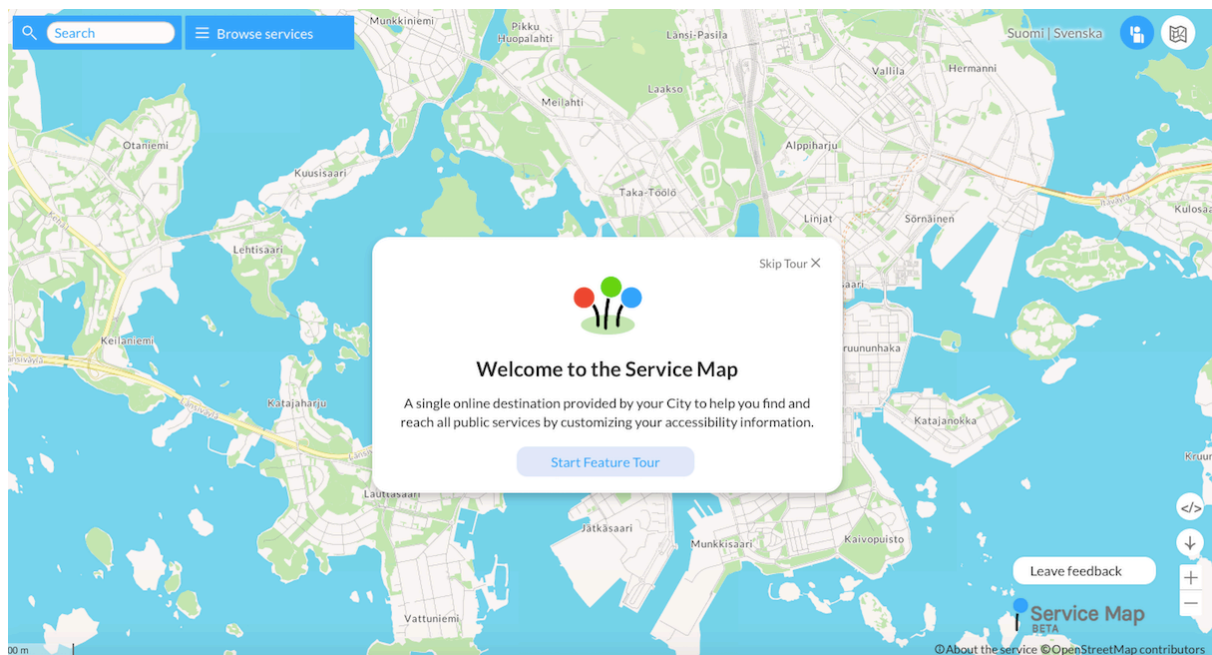
A map server will be also implemented. Map will work as a background to the bartering coop use case. Bartering items are published on the map in geolocations, and they can be commented on, both publicly and privately. Leaflet visualisations can be used to further enhance the visual appearance when there are a plethora of items.

Feature definition: A map that prints geolocation-related ActivityStream data on a map with visualisations.

User Story: As a user, I need to browse items on a map, to find the ones that are close to me.

Description: Map visualises the social media data on a map.

Technical Implementation: Map is a OpenStreetMap with a front-end framework to present the form and commenting fields. Leaflet is used to draw the items on a map as icons and visualisations, depending on the amounts of data in the pilot, and zoom level. The reference implementation for the Map can be seen in <http://servicemap.hel.fi/> and the code in <https://github.com/City-of-Helsinki/servicemap>



2. Integration with other tools

2.1. ActivityStream integration with Maps and Stonecutter


Feature definition: To create a session to enable data reading and writing with single sign on, a Stonecutter integration is needed for the Activitystream datastore and the Map User Interface components.

User Story: As a user, when I start writing to a map form, to fill in a new item in the bartering map, I am prompted to approve using my previously signed stonecutter session, or to log in to Stonecutter.

Technical implementation: Stonecutter API is implemented for both ActivityStreams datastore and the Map UI. The API implements the Open ID Connect protocol and allows applications to delegate their user management to Stonecutter.

Stonecutter demo: <https://sso.dcentproject.eu>
code: <https://github.com/ThoughtWorksInc/stonecutter>

Documentation: <http://dcentproject.eu/wp-content/uploads/2015/08/D5.4-Overview-of-the-OAuth2-Provider-and-Identity-Management-tool.pdf>



Welcome to D-CENT

Connect with all your D-Cent tools.
Sign in securely to applications using your D-CENT Profile Card. You have full control of the information you'd like to share.

Email address

Password

[Forgot password?](#)

Create a Profile

Register and sign in to supported applications using your D-CENT Profile Card.

First name

Last name

Email address

2.2. Crowdsourcing, integration of ActivityStream and Input from Decision articles

The AS2 data is translated into the exact kinds of data (schemas), given in the heterogeneous use-cases given by the partners, by data translation connector applications. In practice small python scripts have been used. The social data-store will enable human-readable metadata.

Feature definition: To create a session to enable data reading and writing with single-signon, a Stonecutter integration is needed for Activitystream datastore and the Decision Article User Interface components.

User Story: As a user, when I start writing to Decision Article forms, I am prompted to approve using my previously signed stonecutter session, or to log in to Stonecutter.

Technical implementation: Stonecutter API is implemented for both ActivityStreams datastore and the Decision Article UI.

Decisions are read from OpenAhjo API and pushed to ActivityStream datastore with Decision Streamer connector developed in 5.1: <https://github.com/okffi/decision-streamer>

2.3. Crowdsourcing, integration of notifications and comments

Feature definition: To enable notifications when comments are being commented on, there is a integration with the ActivityStreams datastore and the commenting functionality.

User Story: As a user, when I log in to Mooncake (the secure notifications tool) I get a new notifications in the feed of the secure notifications tool when my comments are being commented on, so I can contribute in turn.

Technical implementation: Mooncake periodically polls configured sources for new updates. To integrate with Mooncake, the ActivityStreams datastore will expose an */activities* endpoint that list published activities. Mooncake will retrieve activities from this URL and present them to the user, showing the most recent activities at the top of the feed. Users will be able to configure Mooncake to specify which types of activities and which sources they would like to view.

3. Integration Design

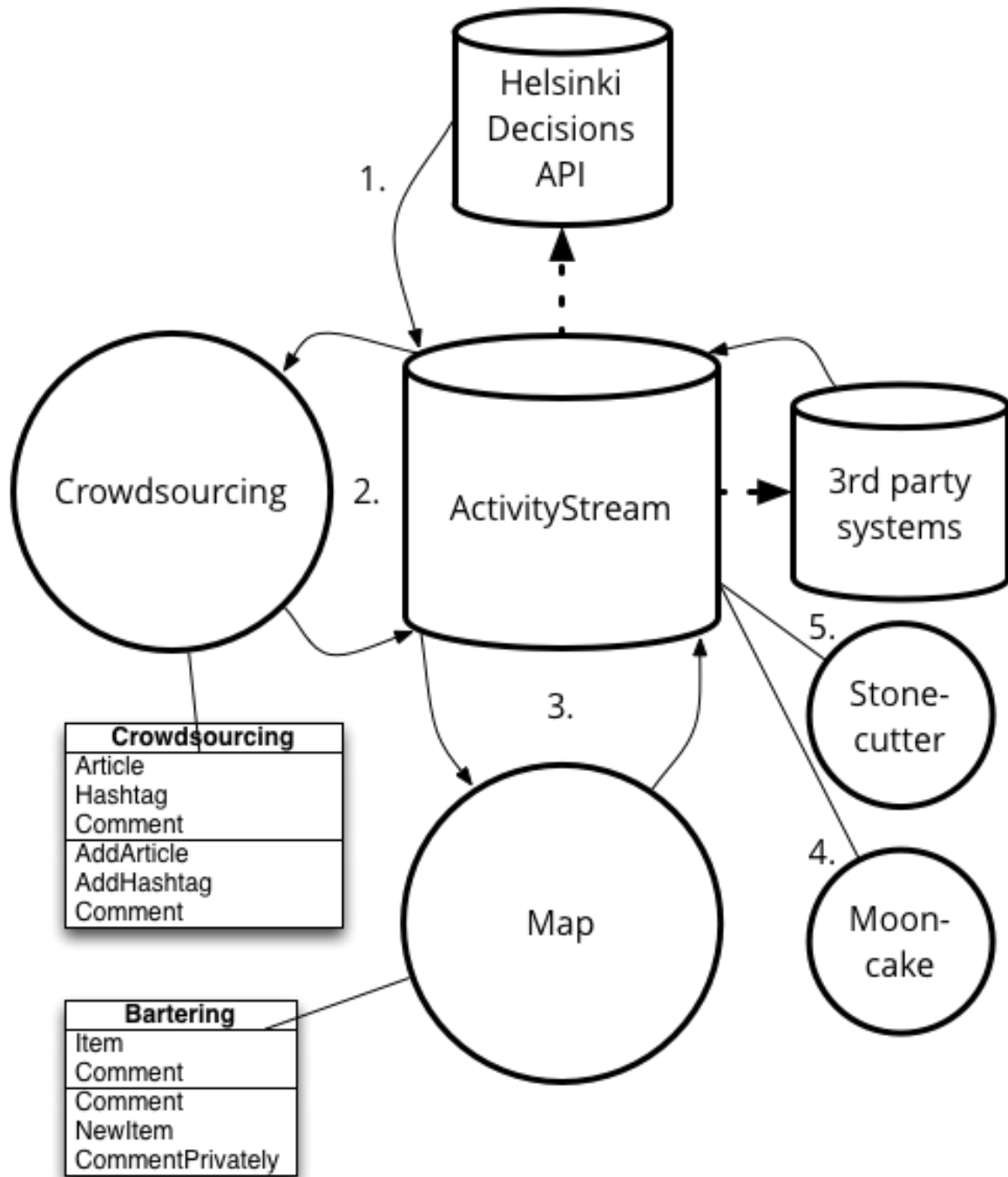


Image 1: Data Cycle with Crowdsourcing and Map Item Bartering

1.	Open Data is read from the Municipality to the ActivityStream datastore.
2.	Data is searched by users. Users open documents and enrich them with comments and hashtags, as well as new versions to documents, and when saved, data is returned to the ActivityStream datastore. The improved keywords and other text is used for semantic matching further search results.
3.	Data for the Map is read from the ActivityStream datastore, new items are added, commented, and written back to the ActivityStream datastore, where it's available for reuse by the local municipality and others.
4.	The ActivityStream is published and consumed by Mooncake, where users can read and configure their ActivityStream feed.
5.	Stonecutter is used for user management. When logging into an application in the platform, the user will be redirected to an instance of Stonecutter to provide their login details, and then redirected back to the application with a logged-in session.

4. User Testing: Finnish Pilot¹

4.1. Pilot progression

In June 2015, a three day workshop was conducted in Helsinki to build up a prototype for local piloting. The hypothesis was that a “pull based” system where citizens can subscribe to decisions they care about will create more online and offline political engagement.

The notion of Social Objects is investigated; “can municipal decisions function as social objects?” Social Objects are the objects of discussion and activity. IMDB has made movies into social objects and Flickr as did the same for photography. In the Helsinki pilot we wanted to test whether municipal decisions - if provided a unique URL and some social features like collaboratively editable hackpad for each decision - could become social objects i.e. the focal points of the discussion.

The problem we identified based on the earlier workshops and user discussions was that people can’t efficiently collaborate and take joint action in reaction to public decisions, if there is no URL that they can go to. Whenever there is a public decision (e.g. to tearing down an old house, raising the salaries of the politicians) that has results in a public reaction, the social media (especially Facebook and Twitter) is filled with comments relating to the decision. However, this discussion remains dispersed on the walls of countless individuals, as it does not link to any place, where like-minded people could meet and, for example, collaborate on an alternative solution or an appeal. If each decision has a URL and if even some people start linking to those pages from the social media discussions, the mechanisms of collective action could start to change.

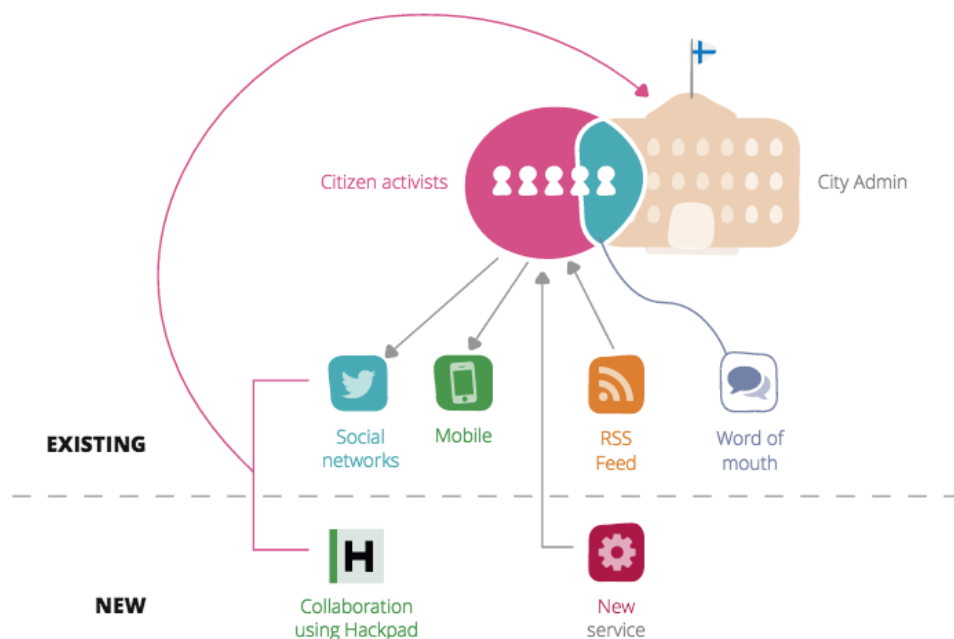


Image 2: Helsinki pilot scenario

¹ See Annex I for a full account of Helsinki user testing

When designing the pilot software the team discussed with local citizen activists to understand their needs and frustrations with the current tools they are using to participate and influence local decision-making.

In parallel to interviews, we also built a basic working prototype of the service and developed it further based on the feedback we received. We identified following key features for the service:

- Simple search
- Results based on search term
- Results are ordered by relevancy to search term most recently published
- Subscription to a term
- Email updates/alerts

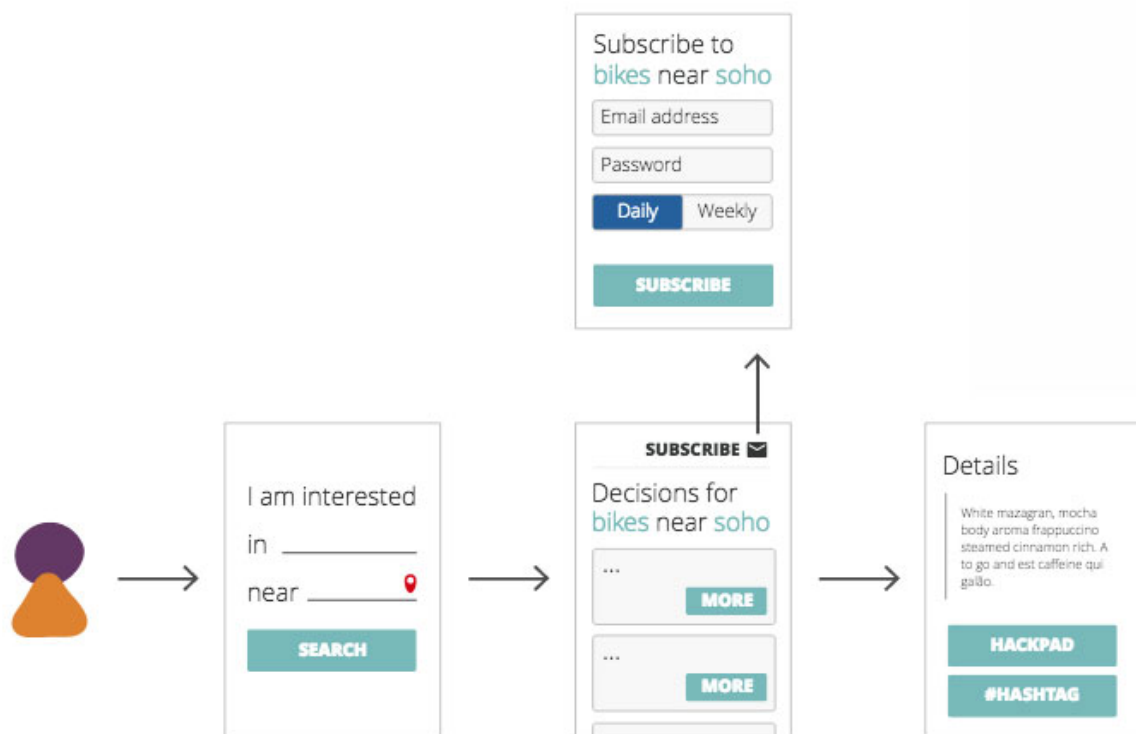


Image 3: Decisions prototype screenshot.

As most of the key features identified were already available, the prototype was actually an integration from already available software, code and tools. This also allowed us to pursue fast and iterative piloting.

The prototype source code resides at <https://github.com/gtrogers/helsinki>.

4.1.1. The Open Ahjo API

The decision-making data utilised for the D-CENT pilot service in Helsinki is fetched from Open Ahjo, an award-winning open interface providing access to all municipal decisions made by the Helsinki city council. Currently, the Open Ahjo interface contains over 40 000 agenda items and over 21 000 issues from more than 8000 meetings.

Open Ahjo was opened in 2013. In terms of transparency, it is a truly unique resource in the world. It has already been utilised in services like [Päätökset](#) (“Decisions”) website (users can browse City Council and committee decisions online), and Ahjo Explorer application (mobile access to the latest council bills and committee verdicts).

In the upcoming years, the hope is that decision-making documentation will be opened up in other Finnish cities as well. As part of [6Aika Six City Strategy – Open and Smart Services](#), Espoo, Vantaa, Turku, Tampere and Oulu.

As part of the follow Helsinki’s lead on opening up decision-making documentation. The plan is to replicate Open Ahjo around Finland – in Espoo, Vantaa, Turku, Tampere and Oulu to be more precise. The work will be done as part of the [6Aika Six City Strategy – Open and Smart Services](#). 6Aika is a strategy for sustainable urban development carried out by the six largest cities in Finland (Helsinki, Espoo, Vantaa, Tampere, Turku and Oulu), between 2014 and 2020.

4.1.2. The Current Version

The screenshot shows a web browser window displaying search results for 'boats' on the Open Ahjo website. The URL is <https://decisions.dcentproject.eu/search?q=boats>. The page features a green header with navigation icons and language options (FI, EN). The main content area has a white background and displays the search results for 'boats' in Helsinki. A prominent heading reads 'I'm interested in boats'. Below this, there is a sign-up form to receive updates on 'boats', with a 'Subscribe' button. The search results are listed below, showing dates and descriptions of decisions.

FRIDAY 05.06.15
Water and street area renting Hakaniemenranta Lucky Boats Oy for the restaurant ship operations
Water and street area renting Hakaniemenranta Lucky Boats Oy for the restaurant ship operations

MONDAY 02.06.14
Change Lucky Boats Oy letting Hakaniemenranta
Renting Area Lucky Boats Oy for the restaurant ship activity Hakaniemenranta

data and mapping module

The current pilot version is running at <https://decisions.dcentproject.eu>. The mobile-friendly service allows users to make easy searches to all the municipal decisions and to subscribe to notifications based on the keywords. On the decision pages the users can share the issues to social media and - importantly - they can open the collaboration pad related to that topic.

Take action

Should this issue be discussed further? Should we co-edit a message to our representatives or the media? Should we prepare a shadow proposal or ask for more information? Use this workspace for collaboration so others, who are interested in the same issue can take part.

[OPEN A JOINT WORKSPACE](#)



4.1.3. The next Steps

As this document is being prepared it is in test use with local citizen activists and regular citizens. Feedback from initial users - some 20-50 users - will be used for further iterating the service before marketing it to the general public.

Based on current findings further development may be needed in the following areas:

- improving the search function (Finnish language optimization)
- allowing users to control their notification subscriptions settings
- packaging the application into a native mobile app to allow better notifications
- improving the hackpad integration potentially branching the software

The software and findings generated in this pilot may prove useful in the future versions of the City of Helsinki's online tool set. The pilot software has been welcomed by the city hall staff and provides useful findings that can help improve services like <http://kerrokantasi.hel.fi> - the online consultation platform of the City of Helsinki and <http://dev.hel.fi/paatokset/> - the decisions portal of the City of Helsinki. The City of Helsinki is currently actively looking to develop new services in the area and the D-CENT pilot has already proved to be well timed to influence the direction new services are being developed.

4.2. ActivityStreams aggregation pilot

In 5.1 we piloted presenting Municipal Decisions data read and consumed from Open Data sources provided by the municipality. In the ActivityStreams data aggregation and crowdsourcing pilot, we are reading data from several Open Data sources, and aggregating it into one D-CENT node's Open Data ActivityStreams datastore, while enriching it with user comments, coedited documents, and hashtags.

As a citizen of Helsinki, when viewing the Municipal Decisions and other Open Data Articles, I want my comments, added hashtags and coedited documents be read back to the aggregated Open Data ActivityStream to be published back to the city.

1. User searches for Articles, and opens one, as described in 5.1.
2. User comments on article, adds a hashtag or contributes to a coediting document
3. User's comment is published back to the ActivityStream as Open Data, and is readable by the Municipality(ies).

4.3. Local refugee co-op pilot

Refugee help coop use case was previously mentioned as the Bartering coop map use case. The pilot group has now narrowed down into a specific situation where local communities want to support refugees' wellbeing through sharing clothing and accommodation.

As a coop member I need to share bartering items in an ActivityStream in D-CENT. As a refugee I need to access on a map what others have shared.

1. User adds a bartering item, with a form that is opened with a button on the coop map: an item or accommodation, to offer. Geolocation is stored and text string metadata is asked on the same form.
2. Description and metadata is uploaded to the service.
3. User is taken to a map of bartering items available with description and written metadata. There should be a radius filter in kilometres.
4. User has a search box on to filter items through category and/or keyword.
5. Users exchange meeting information through private comments.

5. Technology Rationale

Based on the way previous experiments and pilots have turned out, CKAN and Pybossa as external applications are being presumed to be redundant from the perspective of concrete benefits in the context of the pilots in this phase of the D-CENT solution, as it is presumed possible that the benefits expected from CKAN and Pybossa can be delivered by D-CENT core components.

It will also be beneficial not having to create and maintain authentication and data connection integrations to these external tools within D-CENT. Also, technically these tools were built with very different frameworks (i.e. Python-based frameworks as opposed to our current Clojure/Java framework) and so “deep” integration into D-CENT is impossible.

Rather than maintain “stand-alone” platforms such as Pybossa and CKAN, the D-CENT strategy of “loosely coupled components” can be pursued using the existing Finnish APIs such as OpenAhjo. This also fits better into the “Lean UX” approach being pursued by D-CENT.

With more extensive use-specific use cases integrating external solutions is still expected to be relevant in some of the deployments - nodes - of D-CENT. Apart from Pybossa and CKAN, Trello, Jira and Froide have been proposed.

We are reusing results from W3C working group and previous deliverables from Thoughtworks (D5.3; D5.4) as well as Helsinki pilot results to fulfil the needs previously presumed to require external platforms.

6. Security

Existing Helsinki Decisions pilot

The prototype is deployed under HTTPS so that traffic between server and browser is encrypted. SSL termination is performed by Nginx, which subsequently forwards the requests to the application. Secrets for integration with other cloud-based services are encrypted and stored in SNAP CI (a continuous integration pipeline).

Future pilots

The crowd-sourcing and map server pilots will use Stonecutter to establish authentication and access control. Stonecutter is an OAuth 2.0 server that also provides support for the Open ID Connect standard. As well as managing user login, Stonecutter will be used to administer access privileges for members.

These pilots will also integrate with Mooncake (the secure notifications tool in development for D5.6) to publish notifications with integrity. If Activity Streams are signed with JWS (JSON Web Signatures) then Mooncake will verify the signature of the stream to ensure that the data has not been modified in transit.

7. Deployment

Some operations code for deploying the prototype is included in the source repository. Ansible is used in tandem with Docker. The Ansible code is used to provision a remote server with Docker and Nginx. Containers for application dependencies (ElasticSearch and MongoDB) are pulled down and started up as part of the provisioning process. The provisioning process also configures timer tasks (i.e. CRON jobs) for the scheduling periodic updating of the search data from the open data API.

The build artefact is a Python (PIP) package. As part of a continuous deployment pipeline, each new build of the application is transferred to the application server. A shell script is then used to install and start up the application inside a python Docker container.

The application is configured using environment variables. Configuration requirements include secrets for integrating with Mailgun (for automated email sending), and Hackpad (for collaborating on cloud-hosted documents related to decisions).

8. Database Design

Existing Helsinki Decisions pilot

The decisions prototype has minimal data storage requirements. MongoDB is a schema-less JSON document store chosen to store email subscriptions. When a user registers to receive emails for a particular search term, the data is a simple JSON document in the following form:

- {"topic": <SEARCH_TERM>,
- "email":<USER_EMAIL>,
- "unsubscribe_id": <UNSUBSCRIBE_ID>}

The unsubscribe_id is a unique ID used to construct a unique URL that users can use to unsubscribe from a search term.

Future pilots

The future work described above will use an Activity Stream server to aggregate and store Activities. This Activity Stream server will also use MongoDB for persisting Activities. MongoDB is a suitable choice as it is designed for storing JSON documents such as Activity Stream 2.0 objects. MongoDB also provides good support for scaling to use multiple servers, so that aggregation can continue to be performant when the number of integrating Activity Stream sources grows.

The format of the stored documents in this case will simply be the format dictated by the Activity Stream 2.0 specification with any custom vocabularies needed to describe OpenAhjo.