Technical requirements

Decentralised Citizens ENgagement Technologies

Specific Targeted Research Project Collective Awareness Platforms

d-cent

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

Lead beneficiary: ERCIM

D4.2 Technical Requirements and Specification of Platform and Pilots

March 2014
Version Number: 3

Authors:
Harry Halpin
Dennis Roio,
Brian Flanagan

Editors and reviewers:
David Laniado
Sander van der Waal
Robert Bjarnason

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.

Contents

1 Executive Summary

This document outlines the technical specification of the D-CENT project. The approach of the project is not to work from scratch. Building on work that has gone before it will engage with the tools that the pilot projects in Iceland, Spain, and Finland are already using. However, if each pilot technology was improved on independently, there would be effectively three separate applications rather than a unified eco-system that could then be used by others. This unified eco-system will need to communicate via standards as outlined in D4.1 and factor out, from each successful application, a common decentralized social data platform, the D-CENT platform, whose open-source components can then be shared and build future applications. We expect continuous iteration to be necessary throughout the technology development and beyond through use.

What is needed, following the "lean" process outlined in D1.1 and recorded in D1.2, is to build software that users actually want, while taking into account what technical aspects of current applications are currently addressing real needs, as well as a "gap analysis" of where these platforms fall short. Furthermore, across all three pilots common needs should be taken into account, as well as fundamental design principles around data protection, security, and decentralization.

For each pilot to reach success, a MVP (Minimal Viable Product) needs to be created to start this process. While the very first MVPs may be very simple tests of an idea, in order to reach a production-quality solution in the long-term, , the D-CENT consortium cannot be imagined to code a complete solution from scratch, especially given its limited technical resources. Instead, there needs to be a long-term gameplan for integrating the "best of breed" of open-source codebases that fulfil the needs outlined in the social requirement via a technical feature. Thus, in this deliverable we commence a full-scale code-review of all the code-bases used in each pilot, as well as some code-bases identified as relevant from the larger open-source community. This code-bases have been identified by the partners during their creation of the inception events across Iceland, Spain, and Finland. Each of the codebases needs to be analyzed for a variety of factors ranging from their use of open standards, user privacy, and code quality. The main purpose of this deliverable is to analyze each of codebases and then commit a "gap analysis" to where each of them falls short. In final analysis, a number of practical considerations will be taken into account about how open-source code can be re-factored and recombined to fulfil the technical requirements, although for every requirement there will be more development work in the "lean" fashion to make sure that the final result actually does fulfill the needs of real users. While this deliverable will overview the landscape of open-source software projects that could be possibly used by D-CENT and outline how they can be combined in a fruitful fashion, the actual design of the D-CENT platform and each application will happen through the "lean" process in Workpackage 5. Just like the standards recommendations in D4.1, the recommendations in this deliverable are non-binding, but nonetheless provide a valuable map to help coders navigate the disparate and rapidly developing code-bases in the emerging areas of decentralized social networking, direct demoracy, and digital currencies. It is expected that Workpackage 5 will use components of each open-source codebase recommended in this workpackage as the D-CENT platform and each application is designed and built.

2 Social to Technical Requirements

During the inception events in each pilot project, a large amount of interviews with users were gathered, and from these a number of hypotheses were generated using the lean design process. On purpose, we did not ask the users for what "features" they wanted, but what problem they were trying to solve. We then created for each cluster of interviews a "persona" that summarized the main characteristics and problems of a kind of person interviewed. Then, for each problem we developed an initial hypothesis statement for what actions could be taken to address their problem. However, these hypotheses need to be translated into concrete technical features. We consider the hypotheses discovered in the lean design phase to be the social requirements.

It is important to cleanly distinguish social from technical requirements. A social requirement is what concrete social goal we are trying to fulfil in response to the needs of the community, as given by a hypothesis statement. A technical requirement is the software features that we believe will accomplish this goal if the software is adopted. For each hypothesis we gained from the community, we iterate the social requirements in the form of hypothesis statements and then the technical requirements that we believe may fulfil these social requirements. Note that the technical requirements considered here are simply a "first step" and are not considered exhaustive. Many other alternative technical requirements may be manufactured for particular social requirements. Nonetheless, in order to focus software development efforts a concrete technical requirement must be chosen, along with an open-source platform code. This section will simply mention relevant open-source code-bases already used in each pilot, but a thorough review of each open-source codebase will follow later in the deliverable.

2.1 Polling and Proxy Voting

One of the primary needs of the Icelandic user of Your Priorities is that while they could rate the proposals, they could not rate how politicians responded to the proposals: "making it possible to rate responses and actions of politicians/town hall staff to your priorities proposals for cyclists, users of your priorities and active citizens of Iceland we will achieve more pressure on the town hall / parliament to produce higher quality responses."

One of many Icelandic hypothesis statements is: "By testing and improving vote delegation on concrete topics for new parliamentarians, democracy activists and citizens of Iceland we will achieve more trust in and use of vote delegation, making direct democracy tools more viable. We will know this is true when we see results on ease of use and satisfaction amongst constituency of new parliamentarians." The social requirement is the second component, namely that with D-CENT's help "Democracy activists and citizens of Iceland we will achieve more trust in and use of vote delegation, making direct democracy tools more viable."

However, the technical requirement is underspecified in the hypothesis statement, namely that this will require "testing and improving vote delegation on concrete topics for new parliamentarians." Thus, the real technical requirement is that the "Software must feature proxy voting for particular issues." There are a number of open-source code-bases, mainly clustered around the German "Liquid Democracy" project that have a proxy-voting feature. The main code-base that is used by

Icelandic citizens we interviewed is the Your Priorities codebase, it has proxy voting but it is currently disabled as there is no practical strong authentication available, which is a prerequisite for launching any proxy voting features. After proxy voting research by IMMI and reviewing the relevant code-bases, one decision the team in Iceland must make is if resources are available to implement a Minimal Viable Product (MVP) experiment that tests secure proxy voting in Your Priorities.

2.2 Group-based Deliberation

Voting and polling are fundamentally structured activities that are amendable to easy computation implementation. In contrast, many users wanted more of a space for open-ended deliberation before formalizing a particular proposal to vote or issue for a poll. In Iceland, this was phrased as the social requirement of "creating a deliberation space that has a social and informal feel (a place where you can make jokes, share information, express opinions that might not be not fully developed, ask questions in an informal manner)" so that "people feel comfortable sharing unstructured information and undeveloped opinions so new groups can form around these, and develop for them to become issues for action." This ability to discuss informally in well-defined groups can be implemented as deliberation in popular messageboards or using status updates, which serve as the equivalent of messageboards in Facebook and Twitter. However, one requirement for this space of deliberation is the creation of groups, so that deliberation can happen in different sized groups and then new groups can be formed around particular interests. So, one of the technical requirements is group-based deliberation. However, many deliberation boards do not allow linking to background information or even open data. Thus, the social requirement is that they want to add "a structured method of providing background information and references to online deliberation forums" in order to "achieve higher quality/ more informed proposals, more trust in the platform, more chance of scaling the proposals." This seems like it can be technically implemented as having the ability to link to other information outside the platform, as well as sources of open-data. Most platforms allow some form of linking, but the technical requirement is also for some form of "structured" linking, namely linking with some kind of metadata description, possibly as a role within a larger technical process of moving a deliberation, poll, or vote forward. Thus, while any status-update-based codebase such as Status.Net or Diaspora can serve as unstructured deliberation, the kinds of feeds they use must allow an additional level of structure such as that given by ActivityStreams2 (detailed in D4.1), which allow open-ended addition of data and meta-data to the status update.

2.3 High-Value Standardized Authentication

One of the hypotheses from Spain stated that by "creating a shared log-in system for a core set of tools and applications for activists and Partido X, we will achieve more organised and more effective workflows for activists with very little time and resources." What the Spanish MVP wants technically is a shared authentication system, of the kind outlined by BrowserID in D2.1. However, while Mozilla has officially abandoned BrowserID/Mozilla Personae, there is no reason why a shared login system that lets a user authenticate once and then authorize the sharing of their personal data with multiple services using OAuth (and possibly UMA) should be implemented by the D-CENT platform. Although very few code-bases support this, popular libraries such as OmniAuth for Ruby exist for this task. On a more advanced level, a Finnish hypothesis stated that they "believe by collaborating with the ministry to ensure improvement to the national ID authentication system and ensuring it's

compatibility with D-CENT, we will achieve compatibility between Open Ministry and official public service websites, ensuring more trust in online voting and access to public services than when using only user name and password." This particular social requirement for compatibility with a particular national eID system means in general to interoperate with whatever strong authentication system that is used. For example, while Spain does not feature a national eID system, in Iceland the eID system is effectively outsourced to the banking sector. While standards around these authentication systems are still very much maturing (as detailed in D4.1), it seems the technical feature needed is that the D-CENT platform should support using some form of authentication via public key cryptography to be used in any authentication flow. Again, the W3C Web Cryptography API is rapidly maturing to eventually be usable with the hardware-token based systems as well as two-factor authentication (where authentication is checked via a mobile phone using a key on the SIM card) used in Finland and Iceland and we expect a "best of breed" design that consumes national eID identity in parallel with new standardization efforts at the W3C to be deployed in the D-CENT platform.

2.4 Notifications

One of the Icelandic hypotheses is that by "developing a feedback mechanism between representatives and constituencies for new parliamentarians, democracy activists and citizens of Iceland, we will achieve more oversight over actions of democratic representatives, more trust in the system overall." This will require some kind of feedback mechanism in the form of status updates that can be sent to the user when a proposal or other decision reaches a change in state. Another Icelandic persona ended up with virtually the same hypothesis, namely by "incorporating regular updates on the status of a proposal for active citizens in Iceland we will achieve less anxiety over long processing times in the city hall, more participation and trust in the platform." The same hypothesis re-appeared in Finland: "Connecting trials of the DemocracyOS platform with existing open API, generating notifications of town hall agendas, we will achieve better informed citizens, more participation on democracy platforms and more engagement of citizens." A second related hypothesis from Spain is that "creating a responsive web-app for polling for PAH, we will achieve faster decision making in emergency situations." In this case, the fundamental need for a quick feedback status update remains, but the time-scale is different: In the governmental use-cases in Iceland and Finland, the status updates happen in response as feedback to the progress of bills and meeting agendas, while in Spain they happen (often on one imagines a much quicker time-scale) based. This kind of technical feature would require an ability to either create status updates and then push them to users. Technically, this would require a feature that automatically pushes status updates such as ActivityStreams, either within the D-CENT application or more likely a direct push of the status updates via APIs to Facebook and Twitter, and possibly an ability to do both, i.e. create a status update in an autonomous application and push to Facebook and Twitter as needed.

2.5 Status Update Archiving and Customization

There were a number of hypotheses that were grouped around archiving and customizing the status updates. In Spain, one social requirement was that D-CENT create "a relevance and topic based feed aggregator for PAH, we will achieve better knowledge and understanding amongst members of relevant and up to date information that affects their organization." Technically, this requires that the

status updates allow custom topic-based aggregation by categories and then to have the results customized by these categories. Partido X in Spain has the same social requirement, as they want to "create a tool to organise information streams for Partido X, we will achieve better communication internally and between nodes." This social requirement also exists in Finland, as citizens need to navigate the vast amounts of possible meetings in the public sector: "To build a service that allows citizens to 'subscribe' to simple text phrases or tags to notify engaged citizens when topics that interest them are scheduled to be discussed in municipal government." All of these can be done by using a version of ActivityStreams2 that features customizable topics and other arbitrary metadata that currently cannot be found in the status updates produced by Facebook and Twitter. Furthermore, there are social requirements for "creating a method for searching or gathering and archiving past chats, links, posts, images from various platforms in a structured manner" as to achieve a "better record of past activity" in Spain. This will technically require the storage and archiving independently of activity streams, which should come as part of any self-hosted and decentralized solution. There are also innovative social requirements, such as the need of the Spanish pilot for "a real-time translation tool for information feeds for info activists" so that "we will achieve better understanding of and closer collaboration between similar interest-groups and movements across countries and language divides." This can be technically accomplished by extracting the relevant text from an open activity stream and then running it through a translation tool such as Google Translate.

2.6 Open Data Integration

Many of the hypotheses related to features that required some kind of open data facilities. The exact open data features varied by group. For example, in Spain, the PAH group wanted mapping data in order to create "an action mapping tool that identifies main targets and action points for PAH, we will achieve better coordination of actors in response to the threats and more engagement with actions." In other constituencies such as Finland, what was more important was open data about the politicians themselves, as they wanted to make "visible the different voting histories, their campaign promises and political leanings of parliamentarians." Finally, there seemed to be a lack of open data about politics itself, including the procedures in politics, as data was needed to help explain "making constraints (for example planning regulations) and guidelines for a given proposal visible and clear at all times for town hall staff" in Iceland. In general, it seems that there was a wide diversity of open data needed. Technically, the ability to handle generic open data features is required in order to integrate data into decision-making by groups. This can be implemented technically in a variety of ways, ranging from using open data standards such as RDF, or less complex ones such as JSON data available via APIs, as detailed in D4.1. There are many different kinds of APIs and databases that could be used for purpose, such as OpenStreetMaps and CKAN.

2.7 Interoperable Digital Currency Transactions

A number of social requirements around currency were discovered in Spain. In particular, one requirement was that there needed to be a "protocol that settle transactions between users of different websites (tiki, drupal, wordpress, elgg, joomla, cyclos, cclite, ripple, ces, etc.) and between different barter currencies for community." This came up again as the social requirement to allow "users to make interoperable transactions in the barter networks of Catalunya for barter participants in different CES in the region" in order to "achieve a more efficient exchange dynamic

among different local barter networks." In this case, the distinction is not between the different digital sites and their interaction with barter currencies, but between different barter currencies themselves. Technically, the problem facing digital currencies for barter seemed to be a lack of standardized digital infrastructure for the interchange between community currencies and the ability to embed that standard into the many kinds of websites that people use.

2.8 Task Management

One of the over-arching themes of the social requirements is how to transform open data into action. However, most of the previous social requirements involved the integration of open data and structured data into status updates. However, the task of "meta-organization" and assigning tasks to be done in the future is given in the following Finnish social requirement, namely to create "a task management system" so that they can "achieve easier and more effective organisation and less time spent on management." Technically, this requires some form of groupware for citizens to self-organize. However, most groupware solutions such as Trello are currently found only in the proprietary cloud. Thus, we imagine this could be fulfilled by a number of software codebases either in the area of social networking or digital democracy that have some task management component, or a new open-source codebase may have to be created to mimic the features of Trello, or at least integrate a better structured workflow into status updates and existing programs.

2.9 Secure Messaging

Many of the users in Spain had social requirements for a level of integrated security beyond that of Facebook and Twitter. The social requirement was that PAH needed "an integrated cross-platform app with core tools for rapid decision-making, notifications, chat and action maps for PAH" in order to "achieve more effective coordination between PAH nodes, more effective responses to emergency situations and more effective planning of actions." Technically, this requires a level of integration between diverse toolsets generally not available, but one that could be achieved by the use of standards-based interoperability between the various applications. Interestingly enough, one of the social requirements in Spain was given by the hypothesis that "an easy to use secure cross-platform chat system for older activists…will achieve faster and more trusted internal communication between groups." This was demonstrated by the massive number of Spanish users leaving WhatsApp for Telegram. Technically, what is necessary is then the integration of secure messaging across applications, where a high-degree of security using techniques such as encryption for confidentiality and non-repudiable digital signatures. However, the difficult problem is how to create applications that require secure messages while simultaneously remaining usable and without requiring too much additional effort from users in understanding new technology.

2.10 Technical Requirements Summary

The technical requirements, while distinct and often overlapping, do not clearly fit into categories of open source software. For example, some code-bases such as Liquid Democracy feature proxy voting and primitive task management, but do not have the open data features or secure messaging features needed by some applications. However, secure messaging applications such as TextSecure

also lack the proxy-voting and polling features needed by some features. Thus, we have grouped the technical requirements into the following over-arching categories of code-bases:

Social Networking Codebases: These codebases feature some sort of status update based messaging system. One technical requirement that needs to be met are the ability to use feedback status updates in reaction to events and other inputs, and then to send those directly to relevant groups of users. Another technical requirement was that status update customization and search must be possible, so a user can receive only status updates relevant to a particular topic and that they can search both present status updates (and so aggregate status updates) for this topic as well as retrieve previous status updates relevant to a particular topic. These status updates should be integrated across various applications and should feature secure messaging so that activists can trust the codebases. Lastly, the code-bases should feature a single-sign on feature that uses a high-values standardized authentication that can then "plug-and-play" with national identity schemes. This high-level authentication is likely technically important for digital democracy.

Digital Democracy Codebases: These codebases technically feature some kind of polling capability as their baseline, allowing users to poll agreement or disagreement on a wide variety of data. They should also feature some form of proxy voting and a choice of voting algorithms. These code-bases will naturally need to poll and vote over groups, and should feature some kind of unstructured deliberation mechanism as well as the ability to augment verbal interactions and voting/polling with structured linking to relevant data. Finally, in order to transform the democratic decisions into action, some form of software-enabled task management is necessary.

Open Data and Crowdsourcing Codebases: One of the features needed by the D-CENT platform is the ability to work a diverse amount of open data via open data integration into the core platform. In Spain, this took the form of mapping data, while in Finland and Iceland this took the form of data around politicians and political decisions. A large number of structured open data formats could be used and then linked into democratic decision-bodies, and it is also possible that open data may need to be crowd-sourced in order to help make a decision.

Digital Currency Codebases: While there has been an explosion of interest in Bitcoin, it seems the main technical requirement is interoperable digital currency transactions where these currencies may be community currencies around barter networks. Thus, a large number of forks of Bitcoin that allow these digital currencies to be used and bound to particular community currencies should be overviewed.

2.11 Code Review Methodology

Code review is a difficult task, and it is absolutely crucial for D-CENT to build off of the "best of breed" open source products when creating the D-CENT platform and applications. This is difficult to assess in any generic manner, as the requirements can vary. Thus, we will constrict all open-source code review to fit only software that fulfills the technical requirements of the pilots, or software that features a hard dependency in any new code to be created. While some software is very specific, such as "Liquid Democracy" or "Your Priorities," other software such as "Pump.io" or "Apache Wave" may be necessary to create a more generic secure and privacy-preserving decentralized social network that then can be incorporated into the need for citizens in Iceland to have a more accountable government, while also addressing the concerns of the Spanish citizens for

autonomy of data and resistance to surveillance. Furthermore, some software will have to be very low-level and will focus on the new code needed to be developed. For example, one of the harder requirements of the Finnish citizens is to integrate open data into their democratic processes, and the Spanish users also require a way to store the data they have produced collectively.

While all choices will be provisional and may be modified due to the contingencies of development, the goal of this open-source code review is at least get all partners aware of the same code-bases so that development of existing code-bases can proceed and that new code-bases use the correct low-level components rather than replicate effort by using heterogeneous and perhaps even incompatible components. Although the level of ultimate homogeneity amongst all the pilots will never be absolute, as each pilot will have some particular challenges that may not be shared with the rest of the pilots, beginning with a core of shared homogenous code-bases will increase the chance of success. There are some aspects of code review that are very qualitative, including user experience. In order to help judge these, we have included screenshots of software where applicable.

2.12 Licensing

In order for a particular codebase to be used by the D-CENT platform, it must be free and open source software. While there are many closed and centralized solutions to some of the technical requirements, these proprietary solutions will not be used by D-CENT as they cannot be easily modified and so violate the principles of the D-CENT framework. In terms of licensing, this section will list the licensing used by the code-base.

2.13 Programming Framework

This is the programming language the open source codebase has been written in. For example, popular programming languages include Ruby, Python, and Javascript. In general, our approach is to be open to all programming languages but prefer those that are well-known and that are suitable for "lean" development. Also, whether or not standardized frameworks are used is important. As most programming languages have only limited support for the Web, frameworks are often necessary to interact with the Web efficiently. Amongst these frameworks are Ruby on Rails, Django for Python, and Node.js for Javascript.

2.14 Self-Hosting (Data Protection)

European requirements for Data Protection are much more stringent than in most countries, especially the United States. While a full analysis of Data Protection in Europe is outside scope of this deliverable and the new Data Protection Directive is still under negotiation, nonetheless one minimal requirement of the European state-based Data Protection regulations is to allow the data to be hosted concretely in the country. Thus, while we cannot guarantee for each codebase full compliance with European Data Protection regulations (although the D-CENT Platform and applications will do their best effort), we can make sure that each piece of software in D-CENT is capable of being hosted inside Europe. This requires that the software be capable of being installed. For purposes of D-CENT, since we are requiring open-source solutions, we can operationalize self-

hosting as whether or not for any given Web-based piece of software whether or not the software can be installed on a Linux server. While most open-source software is theoretically capable of being installed, for this criterion whether or not a piece of software is capable of being installed will be answered by a "Yes/No" depending on whether or not following the instructions (if they exist) on the website leads the D-CENT consortium to be capable of installing the software. There may also be the possibility of developing heavy client applications, which may be coded in cross-platform C. If a mobile platform is necessary , we will prefer native Javascript but if users need features only available as a native Android application, then the Android Java framework may have to be used.

2.15 Test-Suite

One of the hardest parts of developing software is quality assurance, which is typically done via a test-suite that does unit tests of each component of the software. Open source software varies wildly in their testing, and a sign of a mature code-base is the use of a testing framework. A number of high-quality testing frameworks exist for most modern programming languages, and it is always in general better to use a testing framework rather than simple assertions or a home-grown test-suite. For example, RSpec is the standard test-suite for Ruby. For this criterion, notes will be given.

2.16 Developer Community

One of the more relevant aspects of any open source project is whether or not a vibrant community of developers has emerged around it. While a lone programmer or two can do a large amount of work, the support of interested developers is generally considered a sign of good health for a code-base. The developer community can be described according to many different aspects. One most important aspect is the number of developers actually working on the code-base. However, some developers may be hyper-active while others may commit changes very infrequently. Thus, not only is the total number of developers needed, but the number of commits (changes to the codebase, such as additions of new code) are also an important metric. Lastly, we need to know how long it has been since the last commit, as an open source effort may have once been very active but fallen into dis-use. We plan to monitor this and actively encourage commits from developers.

2.17 Standards Compliance

Many standards were listed in D4.1 as relevant to D-CENT. While many of these standards are immature and full standardization of the suite of social web standards has only just begun by the W3C, if any of these standards are actually used by the particular piece of software will be noted as the answer to this criterion. We will not inspect standards such as HTML5, as we assume that software will be using modern web development, but will instead inspect the standards around social networking, identity, and privacy explored in D4.1.

2.18 Security (Anonymity and Privacy)

The issues of security, anonymity, and privacy of users is of utmost importance to the D-CENT project, and a high level of concern for these issues characterizes the D-CENT platform. However, unfortunately the field of security engineering and privacy-by-design is still relatively new to most open source codebases. The field of designing software that is meant to allow anonymous usage without abuse is even more immature, with not even a textbook existing. Thus, rather unfortunately most open source code bases will not do well in these regards. Nonetheless, unlike closed source software, open source software can be modified and the code reviewed, thus allowing open-source software at least the possibility of using proper security engineering and privacy-enhancing technologies. Although closed-source software may claim to have a high level of security or protect the privacy of its users, due to lack of code review one can never be sure if closed source software does not have an insecure "backdoor" that allows a hacker to attack it or if the software is collecting user data without the user's knowledge, despite any assurances otherwise.

In terms of security, there is no such thing as universally secure software. Software is only secure in lieu of a particular threat model. The threat model outlines the capabilities of an attacker and their goals, and then a security analysis shows how the software prevents the attacker from reaching their goal. Threat models vary wildly: for example, software that may claim to be secure against a global passive adversary with the ability to do focused active attacks, such as an intelligence agency like the NSA, will be vastly different from software secured from an attacker that is a single hacker with limited resources. For the D-CENT platform, we will assume a trusted server (since it is run by the community using commons-based governance) and we will assume that the software itself is not compromised, but installed properly on the user's machine from a trustworthy source. However, we will assume for the threat model of D-CENT targeted attacks on a user account by seizing the password of a user account (or using an automated technique such as Rainbow Tables). The determined attacker may also attempt to seize the user-password database. Thus, the main vectors for attack will be the authentication procedure of the user (so vulnerable to password-attacks, impersonation when another account is seized, or even cooking-snatching attacks to impersonate the user's session). The determined attacker may even attempt to compromise the user-name/password database of the server using open-source off-the-shelf hacking software such as Metasploit (http://www.metasploit.com/) and use services such as CloudCracker (https://www.cloudcracker.com/) to break the (normally MD5) hashing of passwords in the database. Obviously compromising user accounts would be detrimental to use-cases ranging from voting to currency exchange. We will assume as a baseline that standard username/password techniques with a cookie to hold the state of the user (logged in/logged out) will be used by software, and will take special note of any extra security measures taken by the open source software to combat the possibilities of user accounts being hijacked.

Anonymity is more difficult to judge for any codebase. In many places where users may be under threat, it is essential for software to allow anonymity. However, the term anonymity is often confused with pseudonymity, which is simply using a new name (such as "Jaromil" as opposed to "Dennis Roio"). Simply using a new name may give a user a superficial feeling of anonymity, but usually the pattern of use and other data revealed by software is enough to reveal the real identity of a pseudonymous user. These kinds of data include personal data such as explicit data given by age as well as the implicit social network given by friends and contacts. Even if the social graph is not

explicitly listed in the software, observing the traffic, in particular timing information, will reveal the social graph. This kind of signal intelligence, combined with machine-learning, makes true anonymity very hard. Anonymity ends up not in general being a "all or nothing" property, but an information-theoretic measure that is relative to how many other users are possibly sending or receiving a message. Furthermore, for many applications in the D-CENT platform, anonymity may not be a desirable characteristic, such as in voting applications where anonymous voting can easily lead to results being hijacked. However, while it is possible to strongly authenticate without revealing identity information using zero-knowledge proof-based techniques, techniques that have been coded into open-source libraries by the EC Projects ABC4LIFE, these techniques are not used in any open-source codebases. Thus, for the inspection of anonymity, we will assume as a default baseline that a pseudonym is allowed, and note any "real name" policy or stronger authentication (such as national eID schemes) used, as well as any methods to provide more anonymity to users outside of pseudonyms.

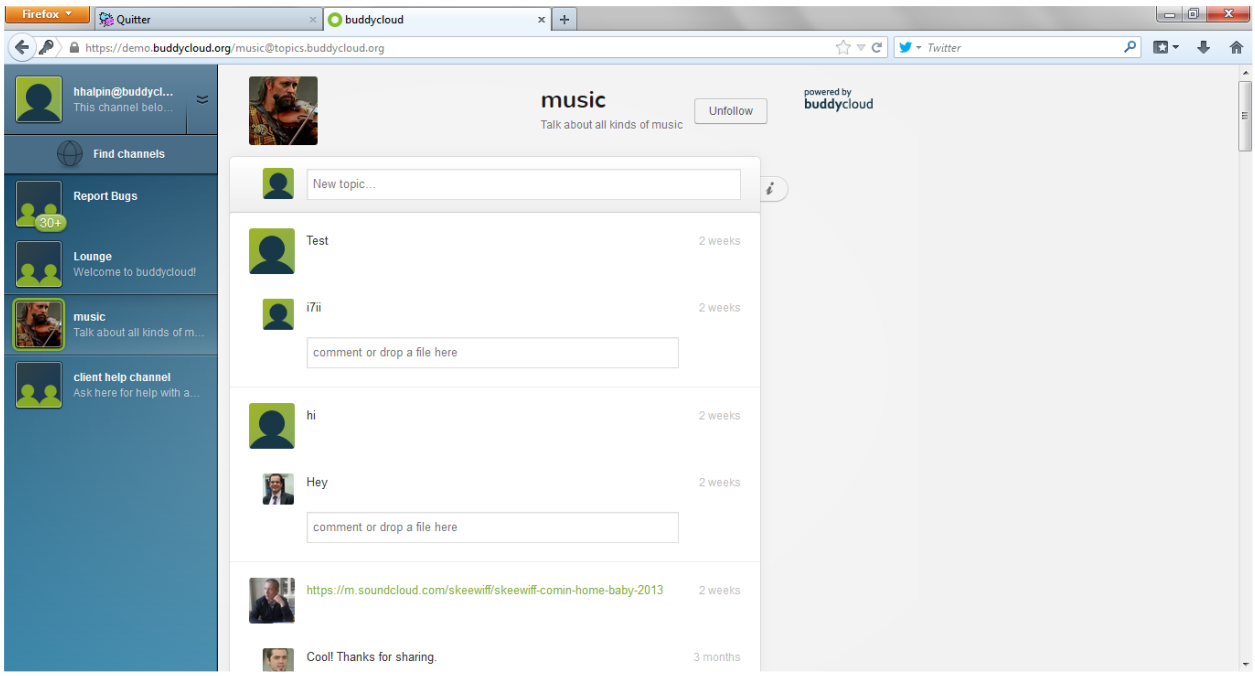
Lastly, privacy considerations should be taken into account. While often these can go by various methods such as involving the user in the consent of having their data being used or alerting the user to possibly privacy invasive techniques (such as done by the eCookie directive) in general the D-CENT project will assume that the self-hosted D-CENT nodes will do their best to adequately describe their terms of service to the user. Instead, we will focus on data minimization, which is only retrieving personal data from the user when necessary. We will assume that some username and login is required. However, if the open source software takes more data from the user, the D-CENT platform should be able to display precisely what data it has on a particular user or group of users and given the user the option of downloading their data (data portability) in a standardized format, correcting incorrect or misleading personal data, and removing data from a server. These provisions are in-line with the proposed EC Data Protection Directive. We will assume as a baseline that the software does not do anything to minimize data collection or offer the user control over their own data, and will note if any codebase does allow these capabilities.

3 Social Networking Codebases

The core of the D-CENT platform should be a shared baseline of functionality for D-CENT nodes to allow them to communicate in a decentralized and federated manner. A D-CENT node is simply any application that can communicate with other D-CENT nodes using the standardized interfaces. The D-CENT platform can then be thought about as a lightweight group of shared methods around sharing messages, profile data, social graphs, authorization, and authentication between D-CENT applications. As outlined in D4.1, this must involve the use of messaging standards that allow D-CENT nodes to send each other, with the goal being to establish a decentralized social network whose social capabilities can be accessed by any D-CENT application. There has been a rash of decentralized social networking codebases in response to the centralization of user data in closed silos such as Twitter and Facebook, and this section reviews them in light of the technical requirements of D-CENT. We only include explicitly social messaging tools here that can fulfil some

form of status update and notification requirements, with tools more based on task management and collaborative document editing being inspected for integration in D4.3.

3.1 BuddyCloud

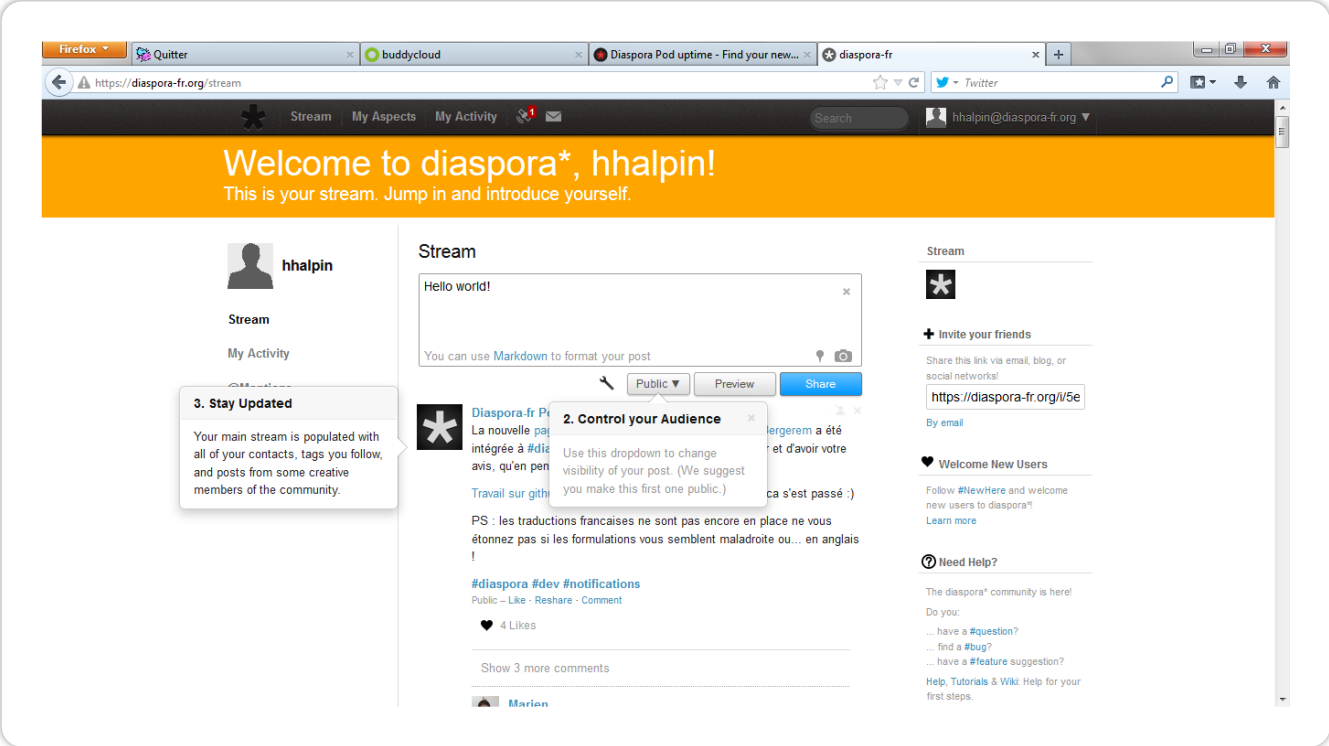


Name	BuddyCloud
Website	http://buddycloud.com/
Code	https://github.com/buddycloud
Licensing	Apache2
Programming Framework	Java for client, Javascript and node.js for Web-facing parts
Self-Hosting	Yes
Test Suite	Yes, although ad-hoc and in Python for server and using Jasmine for Javascript

Developer Community	16 developers, 2000+ commits, daily updates
Standards Compliance	XMPP
Security, Anonymity, and Privacy	The codebase uses standard techniques for protecting connections between servers-to-servers and server-to-client in XMPP for forcing TLS connections, encrypting all data while in transit although not on the server.

The BuddyCloud federated social networking platform is an active codebase, using XMPP to build a federated Twitter-clone. However, no other servers except the central one use their codebase and they are not compatible with other efforts. D-CENT would only build from their codebase if XMPP was chosen as a substrate. They have no anonymity or privacy considerations outside of self-hosting, and their security considerations are nothing outside of standard XMPP security considerations.

3.2 Diaspora

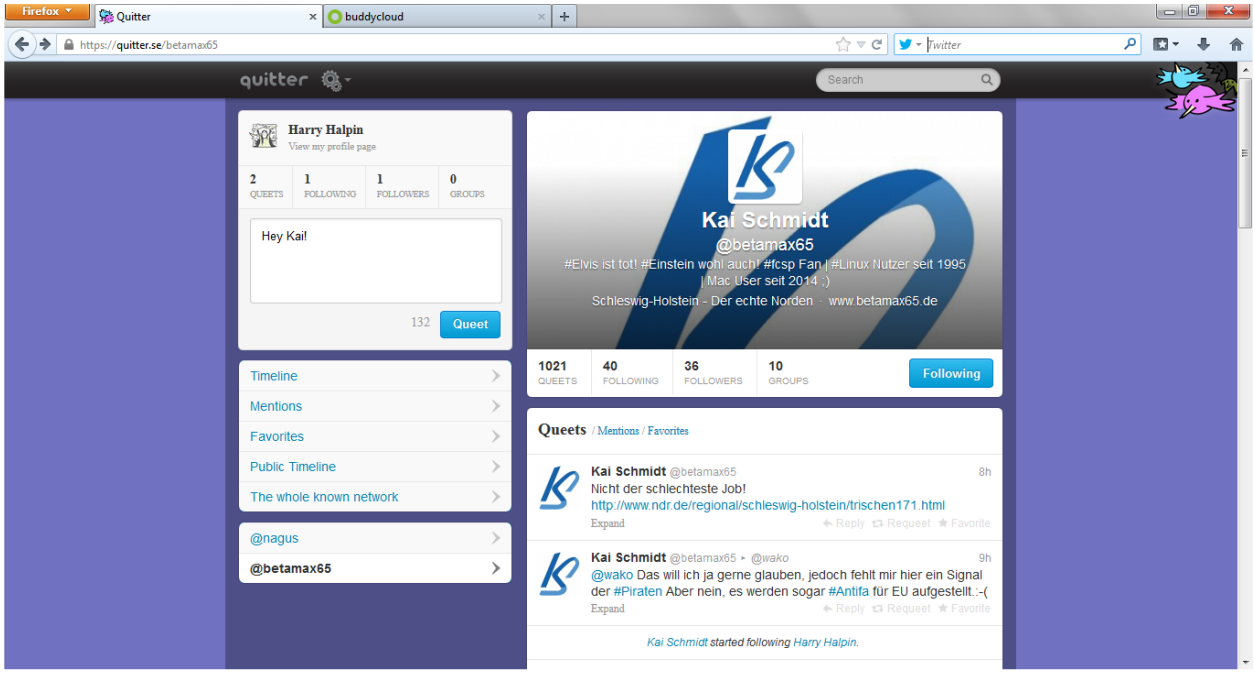


Name	Diaspora

Website	https://joindiaspora.com/
Code	https://github.com/diaspora/diaspora
Licensing	AGPL 3.0
Programming Framework	Ruby on Rails
Self-Hosting	Yes
Test Suite	RSpec, in process
Developer Community	261 programmers, 10,000+ commits, daily activity
Standards Compliance	OStatus (Pubsubhubbub, ActivityStreams), Vcard (via hCard), ActivityStreams, Salmon Protocol.
Security, Anonymity, and Privacy	Although there is a dedicated security bugzilla, they still seem to be fixing basic security bugs. However, there is active discussion of increased use of encryption in messaging and better privacy, although no real consideration

Diaspora is the famous open source "clone" of Facebook. Although the project nearly died when the founders left it, Diaspora now has a vibrant open source community that is trying to resurrect the project. Its use of Ruby and many open standards make it a suitable building block for D-CENT, although it still needs improved testing, standardization compliance, and privacy/anonymity considerations, and in many respects may be too heavy-weight for some of the D-CENT use-cases. A list of deployments of Diaspora can be found here: http://podupti.me/.

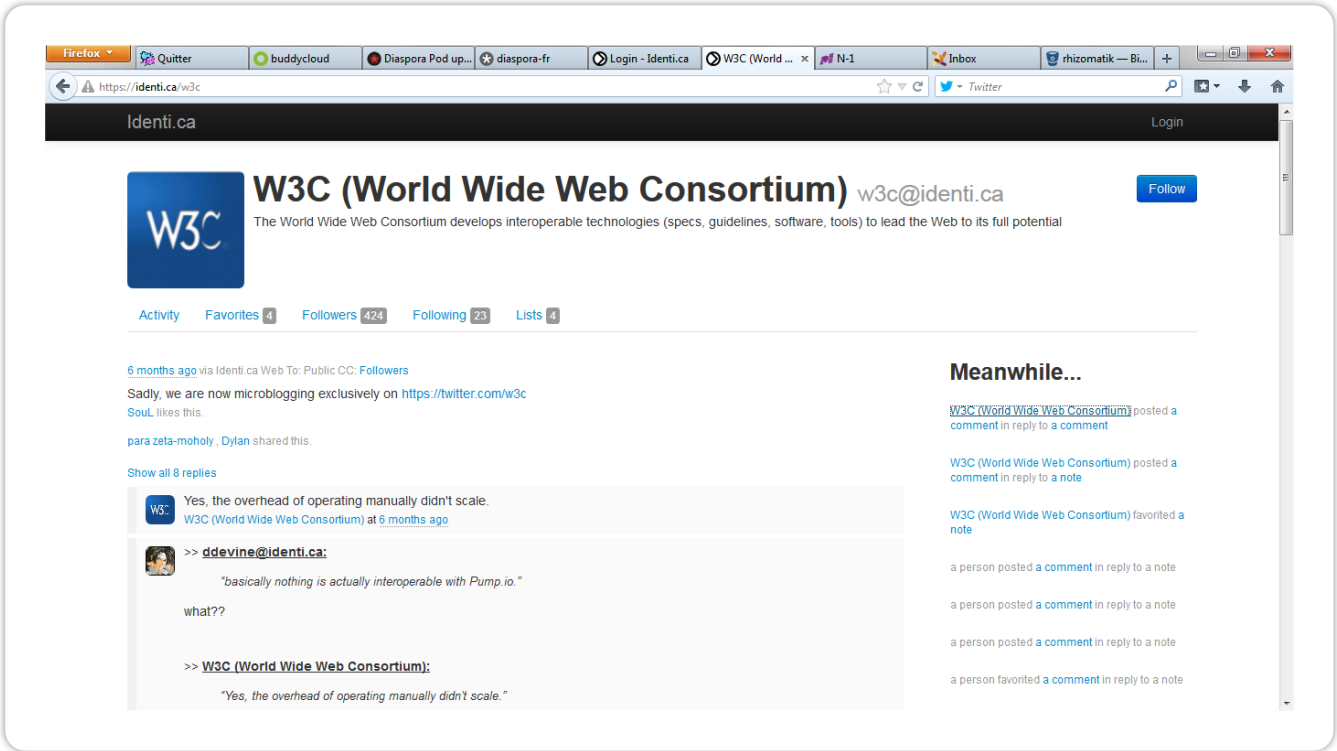
3.3 Status.Net



Name	Status.Net/GNU Social
Website	http://status.net/
Code	https://github.com/zh/statusnet
Licensing	AGPL 3.0
Programming Framework	PHP
Self-Hosting	Yes

Test Suite	A few assertion-based testing
Developer Community	Formerly active, now dead. 10,000+ commits, 52 developers, but no commits for last two years.
Standards Compliance	OStatus (Pubsubhubbub, ActivityStreams), Vcard (via hCard)
Security, Anonymity, and Privacy	No regard outside normal Web practices and hobbled by generally insecure nature of PHP.

Although the lead programmer Evan Prodromou has abandoned this project to create a standards-based open-source Twitter clone, the Free Software Foundation has taken it over and re-branded it GNU Social. Nonetheless, there are almost no installed instances and the codebase has gone completely moribund in terms of activity. The PHP/MySQL framework also encountered scaling problems. Thus, it is unlikely D-CENT will re-use any of the code, although its standards-based approach is exemplary.

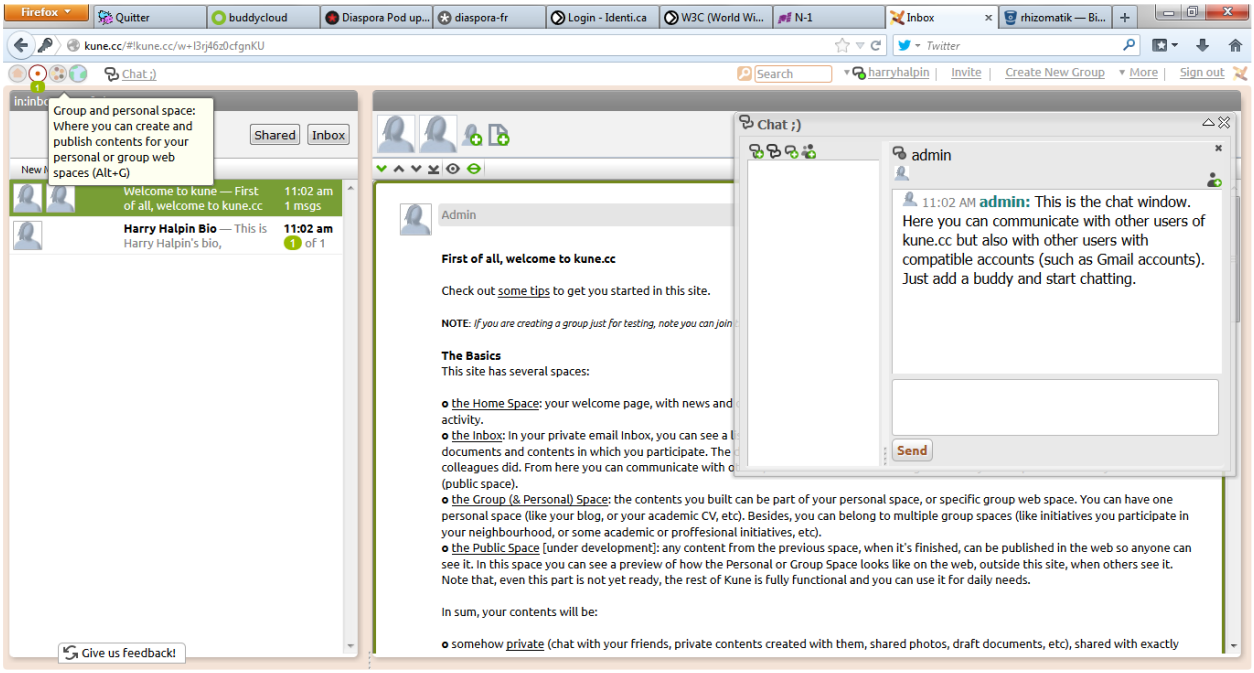


3.4 Pump.io

Name	Pump.io
Website	http://pump.io/
Code	https://github.com/e14n/pump.io
Licensing	Apache License 2.0
Programming Framework	Javascript (Node.js)
Self-Hosting	Yes
Test Suite	Some tests in code but not formalized
Developer Community	Approximately 3,700 commits, 9 developers, monthly activity
Standards Compliance	ActivityStreams, OAuth.
Security, Anonymity, and Privacy	Security updates (XSS vulnerabilities noticed) but not much beyond standard best practices on the Web. No special considerations of anonymity and privacy.

Pump.io is the next-generation of Status.Net, a Twitter clone, re-written as a more high-performance and general native Javascript version meant to eventually be OStatus compliant. For example, Identi.ca transferred from Status.Net to Pump.io. However, while the codebase has made some progress, it is still far from complete and has a lack of an active developer community. As regards some native Javascript code for ActivityStreams could be re-used.

3.5 Kune (Apache Wave)



Name	Kune, a fork of Apache Wave – formerly Google Wave.
Website	http://kune.cc/ (http://incubator.apache.org/wave/)
Code	https://gitorious.org/kune (http://incubator.apache.org/wave/source-code.html)
Licensing	AGPL 3.0
Programming Framework	Java and Javascript (via Google Web Toolkit)
Self-Hosting	Yes
Test Suite	Partial
Developer Community	5 developers, approximately 2,500 commits, daily activity. Due to not using Github, activity on Apache Wave is difficult to judge, but seems non-existent.
Standards	XMPP and non-standardized Google Wave Protocol

Compliance	
Security, Anonymity, and Privacy	Standard XMPP security, although not updated to deal with latest XMPP work or Web Security model.

Kune is a re-branding with additional features of Google Wave, which Google abandoned in 2010 and gave as open-source code to the Apache Wave foundation. Since then, a new Spanish developer community has re-invigorated the project and almost doubled the size of the codebase, which now is used by the EC project P2PVALUE and has impressive features. However, the use of deprecated Google work like Google Gadgets and the choice of Java as programming language make re-use of this codebase by D-CENT difficult. Interoperability via upcoming W3C Social Web standards is more likely to work out.

3.6 Elgg (Lorea)

Name	Lorea (Elgg)
Website	https://n-1.cc/ (http://elgg.org/)
Code	https://gitorious.org/lorea/ (https://github.com/Elgg/elgg)

Licensing	AGPL 3.0 (GPL v2.0/MIT)
Programming Framework	PHP
Self-Hosting	Yes
Test Suite	Uneven
Developer Community	Not using Github so difficult to tell. For Lorea, small developer community (around a dozen) and monthly updates. For Elgg, 10,000+ commits, 50 developers, weekly updates.
Standards Compliance	OpenID Connect, ActivityStreams,
Security, Anonymity, and Privacy	OpenPGP implemented and some security considerations taken into account, but hobbled by generally insecure nature of PHP.

Lorea is a fork of the Elgg decentralized social networking codebase used in Spain, with many additional features needed by social movements. While there was a massive jump in users during the M15 social movement, the codebase experienced scalability and maintenance issues, so use went down. Lorea also tried to implement many W3C standards, including most of OStatus, on top of Elgg. However, the developer community has due to security concerns now mostly moved to other projects and the user community seems frustrated. Although Lorea was ahead of its time in ternms of thinking, like Status.Net, it is highly unlikely that D-CENT will re-use much of this codebase given its PHP framework and declining developer community.

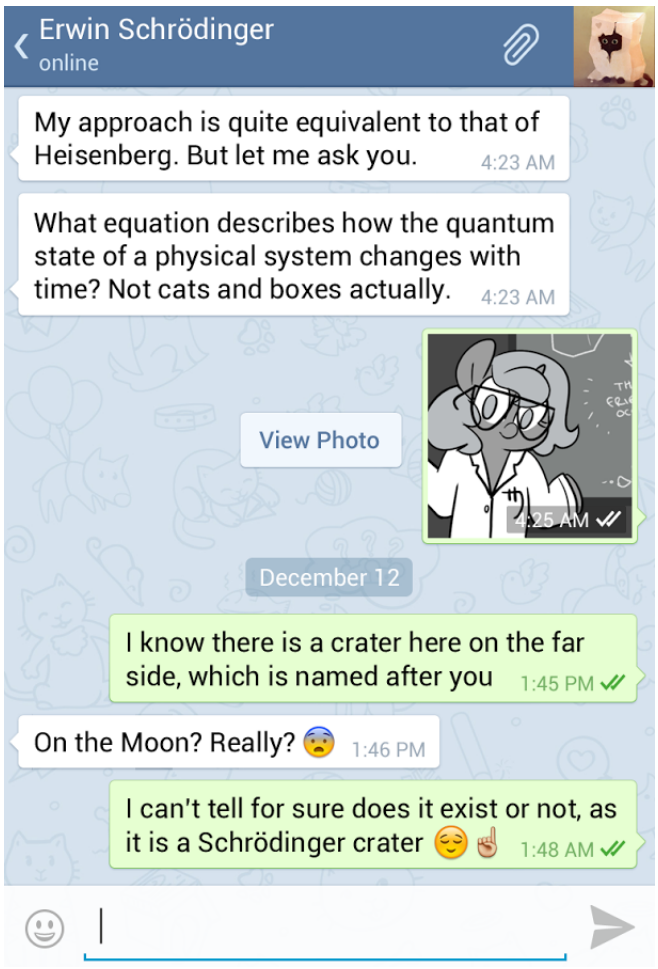
3.7 ComPAHS

Name	ComPAHS
Website	https://play.google.com/store/apps/details?id=kamis.compahs
Code	Not available yet
Licensing	Unknown, but base programming language is non-open source.
Programming Framework	Delphi
Self-Hosting	Yes
Test Suite	Unknown

Developer Community	Lone developer, new software.
Standards Compliance	Unknown
Security, Anonymity, and Privacy	Likely to be very little.

This application is actually a "meta"-application developed to tie together the functions of chat, messaging, decision-making, chat, and mapping for the PAHS anti-eviction campaign. It features many innovative features such as the mapping of new local groups and housing actions. However, it is coded by a single coder in the Delphi (a variant of Pascal) framework, a programming language that is effectively moribund and not open-source. Thus, D-CENT can learn from the general framework but cannot re-use the components due to the choice of programming language and licensing concerns.

3.8 Telegram

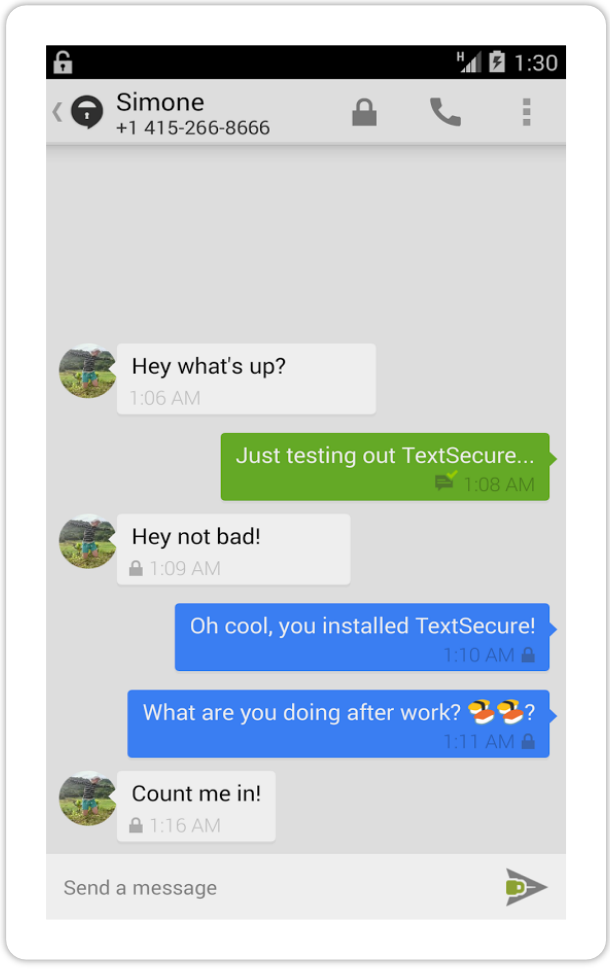


Name	Telegram
Website	https://telegram.org/
Code	https://github.com/DrKLO/Telegram
Licensing	GPL v2
Programming Framework	Java (Android)
Self-Hosting	Yes
Test Suite	Uneven

Developer Community	6 developers, 50 commits, updated mostly monthly
Standards Compliance	Works with mobile standards, no Web standards. Non-standard, created own protocol MTProto.
Security, Anonymity, and Privacy	Claims to be secure and privacy/anonymity aware, but actually is insecure. They created their own non-standard cryptographic primitives, which have been shown to be insecure. See here: http://hn.meteor.com/posts/6913456-141ae

After Facebook acquired WhatsApp, this start-up by the Russian founders of social network VKontakte gained millions of users with its claims of "end-to-end" security. Unfortunately, the code uses their own "cryptography" and has had vulnerabilities discovered. Although the developer community seems happy to fix vulnerabilities, they clearly have little concept of security but are simply advertising as such. Thus, it is highly unlikely D-CENT will build from Telegram.

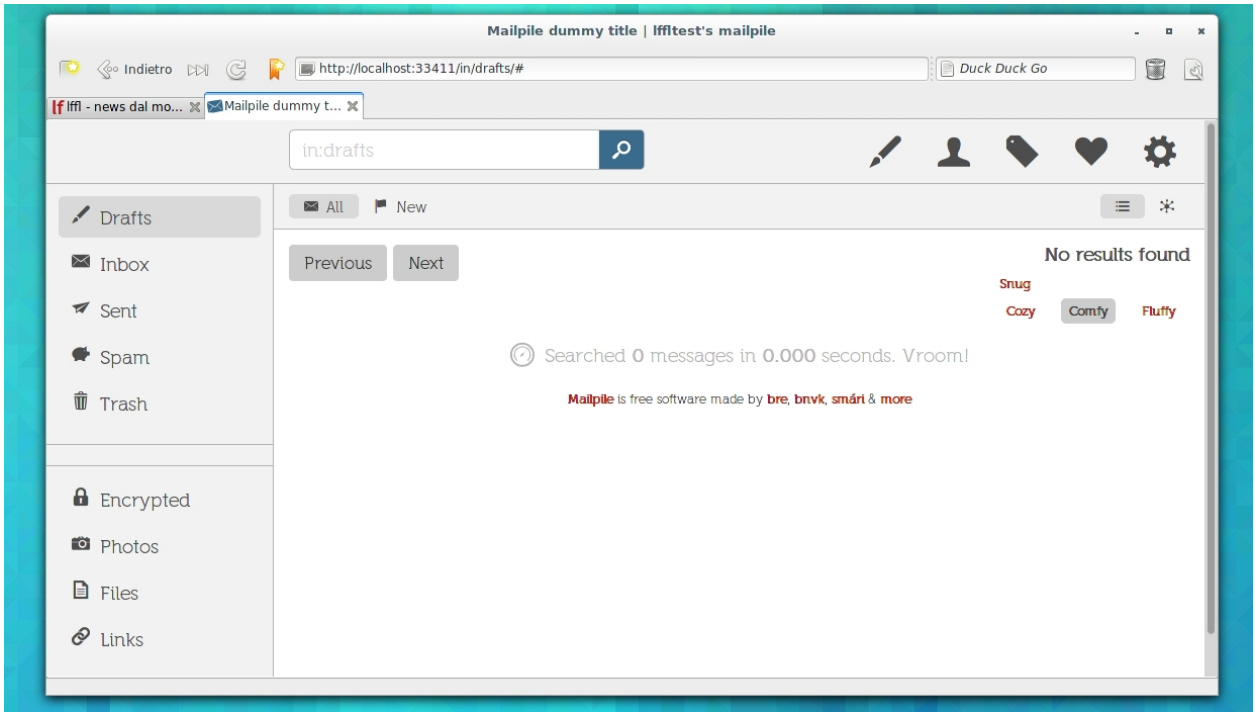
3.9 TextSecure



Name	TextSecure
Website	https://play.google.com/store/apps/details?id=org.thoughtcrime.securesms
Code	https://github.com/WhisperSystems/TextSecure/
Licensing	GPL v3
Programming Framework	Java (Android)
Self-Hosting	Yes
Test Suite	Yes
Developer Community	Approximately 760 commits, 50 developers, monthly commits
Standards Compliance	Works with mobile standards and standard Off-the-Record messaging, but no Web standards. Developing possible standards for group chat based in public on wiki.
Security, Anonymity, and Privacy	Very high security and privacy considerations. Data stored encrypted while in transport and locally. Several improvements to OTR protocol, including the advanced Axolotl key ratchet to provide forward secrecy.

TextSecure features very mature secure messaging over the Internet. It takes advantage of Android capabilities to do secure applications where the data is encrypted locally and encrypted even on the server. However, it does not work as Web application precisely for these reasons, only as a mobile application.

3.10 Mailpile



Name	Mailpile
Website	http://mailpile.is
Code	https://github.com/pagekite/Mailpile
Licensing	GPL 3.0
Programming Framework	Python
Self-Hosting	Yes
Test Suite	Began
Developer Community	60 developers, 1900 commits, daily updates.
Standards Compliance	SMTP (Email)

Security, Anonymity, and Privacy	OpenPGP (GnuPG)

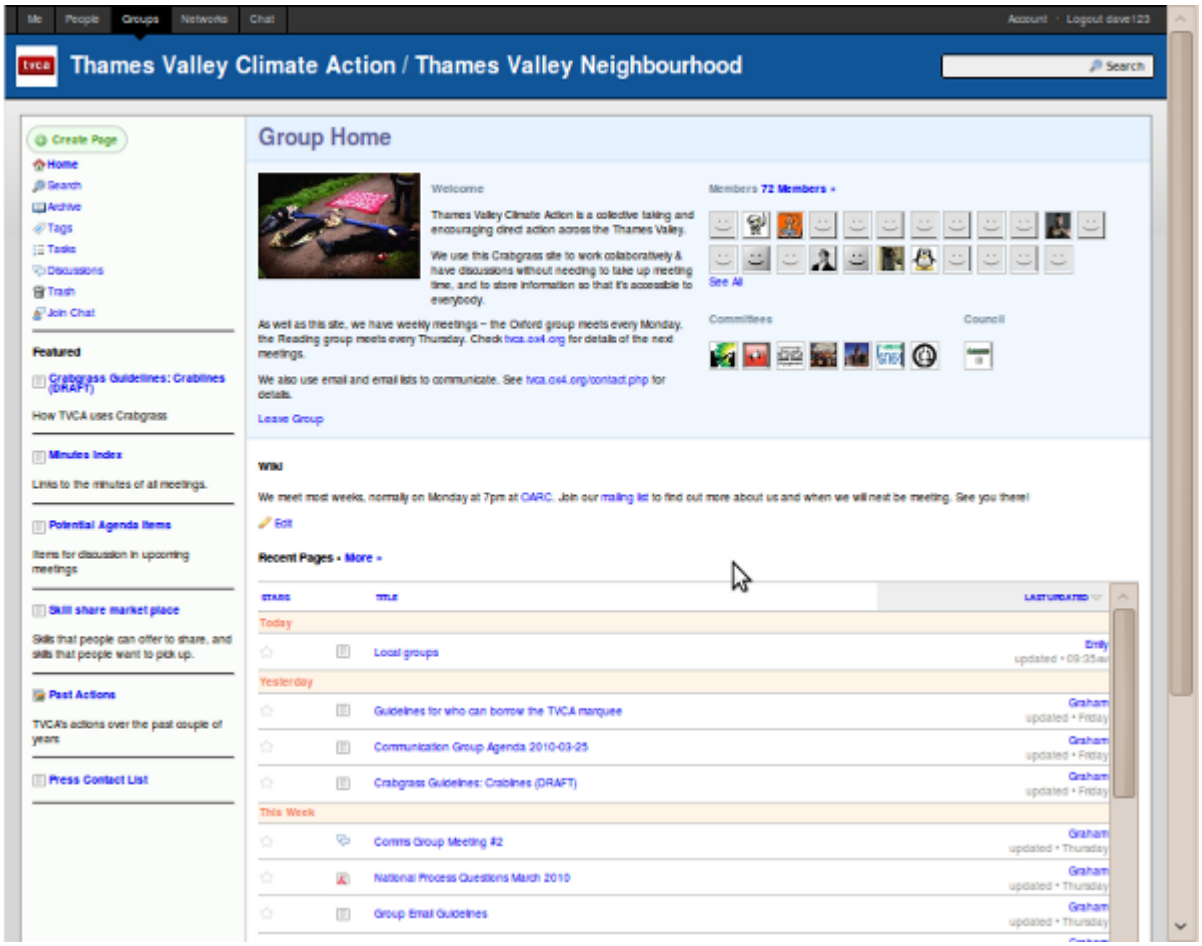
Started with much fanfare due to a Kickstarter campaign, Mailpile is meant to be an email client. Current open source e-mail clients like Thunderbird for the most part fail to search mail properly, and Mailpile hopes to search e-mail on the client as easily as Gmail. However, Mailpile has just started and so is not yet ready for use, and is currently limited to using secure messaging only a single single device.

3.11 LEAP

Name	LEAP Encryption Access Project
Website	http://leap.se
Code	https://github.com/leapcode/
Licensing	GPLv3
Programming Framework	Python
Self-Hosting	Yes
Test Suite	No
Developer Community	10,000+ commits, 12 developers, commits daily
Standards Compliance	SMTP, (E-mail), and other IETF (Internet Engineering Task Force) related email standards (DKIM, Certificate-pinning)
Security, Anonymity, and Privacy	Private keys are stored in an OpenVPN that runs on the client and never exposed to the server. All passwords needed to bootstrap installation use Secure Remote Password. Employs most security-related email standards such as DKIM and S/MIME

LEAP is an open-source project to develop encrypted email, with a focus on client-encryption, multiple multiple-devices and securing the server using a variety of standards and techniques. Similar to TextSecure, with LEAP even the server should not be able to read the email. However, the software is still under development and may need to be combined with the Mailpile work to reach a full, user-friendly solution.

3.12 Crabgrass



Name	Crabgrass
Website	https://we.riseup.net/crabgrass
Code	https://github.com/riseuplabs/crabgrass
Licensing	AGPL3.0
Programming	Ruby

Framework	
Self-Hosting	Yes
Test Suite	Uneven
Developer Community	19 developers, 6,333 contributors, inactive since 2012
Standards Compliance	Some work on XMPP
Security, Anonymity, and Privacy	Focus on security and privacy via access control, but no federation due to security concerns.

Crabgrass is a project to create a secure, self-hosted social network aimed at activists. It features complex group control and co-edited wiki documents, but it does not feature federation due to security concerns, although there were some XMPP-based experiments. Activity seems to have died down as most core developers moved to work on LEAP.

3.13 GNUNet

Name	GNUNet
Website	https://gnunet.org/
Code	https://gnunet.org/subversion
Licensing	GPL3
Programming Framework	Reference implementation and library in portable C. A reimplementation exists in Clojure (Java)
Self-Hosting	Yes
Test Suite	No

Developer Community	9 core developers, more than 50 contributors including translations and main distribution maintainers.
Standards Compliance	GNUnet is a P2P protocol on its own. Reference implementation is POSIX C and library dependencies are commonly present and portable.
Security, Anonymity, and Privacy	GNUnet claims very strong anonymity and privacy. GNUnet configures itself as a next generation P2P network with emphasis on anonymity although there are known attacks (http://freehaven.net/anonbib/cache/kugler:pet2003.pdf)

GNUNet is a peer-to-peer network protocol that could be used for some social networking and messaging functions. However, being done in Portable C, it would be more naturally suited towards heavy-clients (and thus the digital currency software) than a web-based one. See D4.1.

3.14 Twister

Name	Twister – P2P microblogging platform
Website	http://twister.net.co/
Code	https://github.com/miguelfreitas/twister-core
Licensing	BSD/MIT
Programming Framework	C++, based on the BitcoinD reference implementation.
Self-Hosting	Yes
Developer Community	One talented lead developer, followed by a good community of interested people. The project is quite popular in the Bitcoin community http://tech.slashdot.org/story/14/01/07/161257/twister-the-fully-decentralized-p2p-microblogging-platform
Standards Compliance	BitcoinD implementation + libbittorrent to support DHT message distribution.

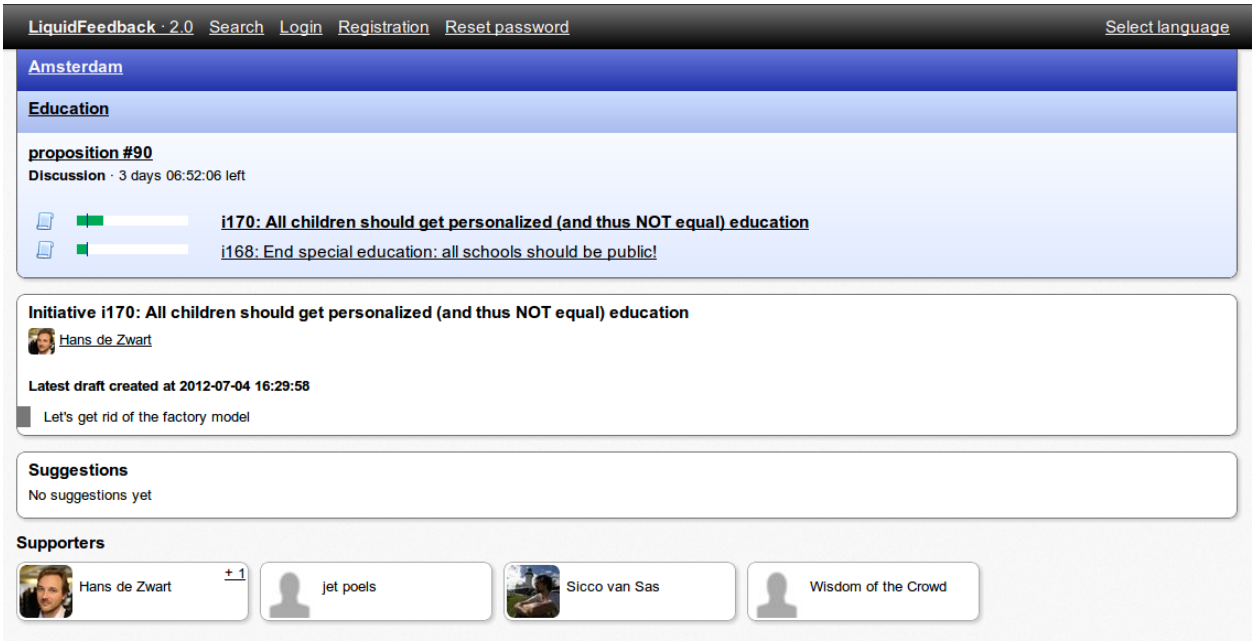
Security, Anonymity, and Privacy	Inherits characteristics from BitcoinD (see section 7). Twister is 100% decentralized.

Twister is interesting to look at for its approach to P2P (DHT) distribution of a blockchain as well for the focus on short (tweet-like) messaging. It may be useful to interface as backend for Web-facing pilots and the digital currency pilots.

4 Digital Decision-making Codebases

These codebases feature polling and proxy voting. They are quite essential to citizen engagement, but in general the landscape is even more diverse than that of social networking codebases. Most of the codebases feature only primitive social networking and notification functionality. However, there are many codebases and the space is moving very quickly.

4.1 Liquid Feedback

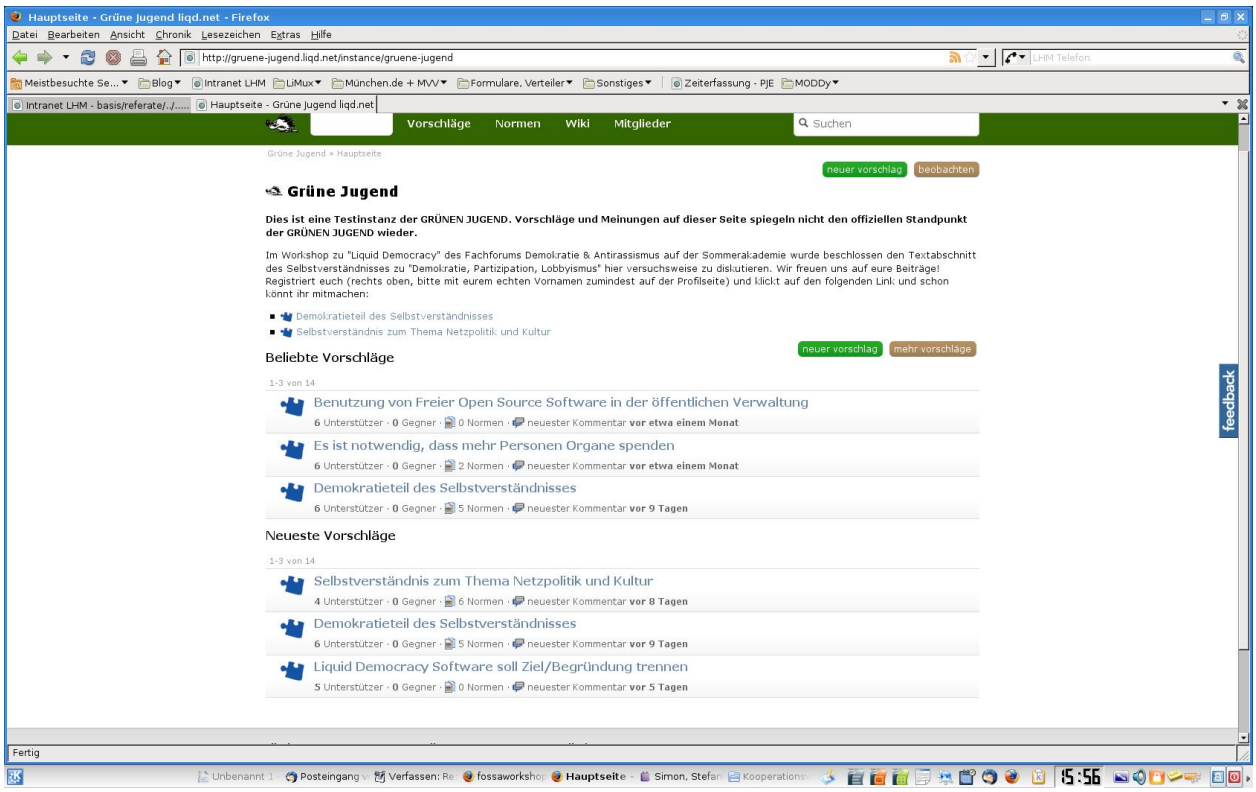


Name	Liquid Feedback
Website	http://www.liquidfeedback.org/

Code	http://dev.liquidfeedback.org/trac/lf/
Licensing	MIT
Programming Framework	Lua, C, WebMCP
Self-Hosting	Yes
Test Suite	No
Developer Community	Minimal: 2-3 semi-active developers via Public Software Group. Not using github.
Standards Compliance	Minimal: non-standard (but well documented) public API
Security, Anonymity, and Privacy	Best Web practices

The original proxy-voting codebase used by the German Pirate Party. It was ground-breaking at the time, but features a difficult to use user-interface.

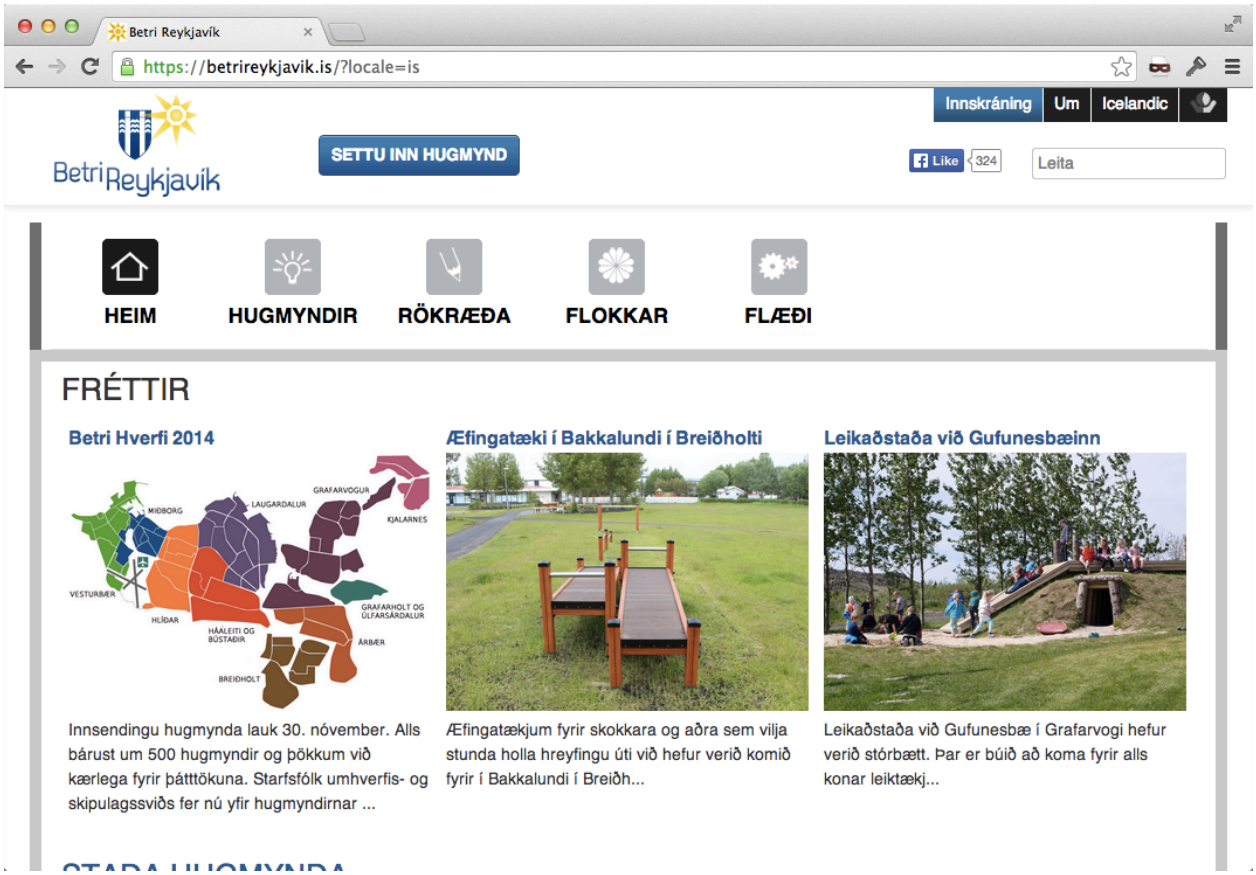
4.2 Adhocracy



Name	Adhocracy
Website	https://liqd.net/en
Code	https://github.com/liqd/adhocracy
Licensing	GNU AGPL v3
Programming Framework	Python 2, Pylons
Self-Hosting	Yes
Test Suite	Good coverage, continuous integration
Developer Community	Good: 26 contributors, 5600+ commits, daily updates
Standards Compliance	No public API
Security, Anonymity, and Privacy	Best Web practices

A Python rewrite of Liquid Democracy, featuring a better user-interface and more standard best practices for the Web.

4.3 YourPriorities



Name	YourPriorities
Website	https://www.yrpri.org/home/world
Code	https://github.com/rbjarnason/your-priorities
Licensing	GNU AGPL v3
Programming Framework	Ruby on Rails

Self-Hosting	Yes; via Docker
Test Suite	Test Suite: One browser based test testing all key features with browser automation
Developer Community	2 contributors, 330+ commits, daily updates
Standards Compliance	No public API
Security, Anonymity, and Privacy	Best Web practices

Formerly called "Social Innovation" this codebase features polling, debate and deliberation features. It has one of the largest user-bases of any digital democracy codebase. It's been giving citizens of Reykjavík real influence since 2010 and two laws in Estonia have been approved by the Estonian parliament that originated using Your Priorities. With additional users being piloted in NHS Citizen project in the UK.

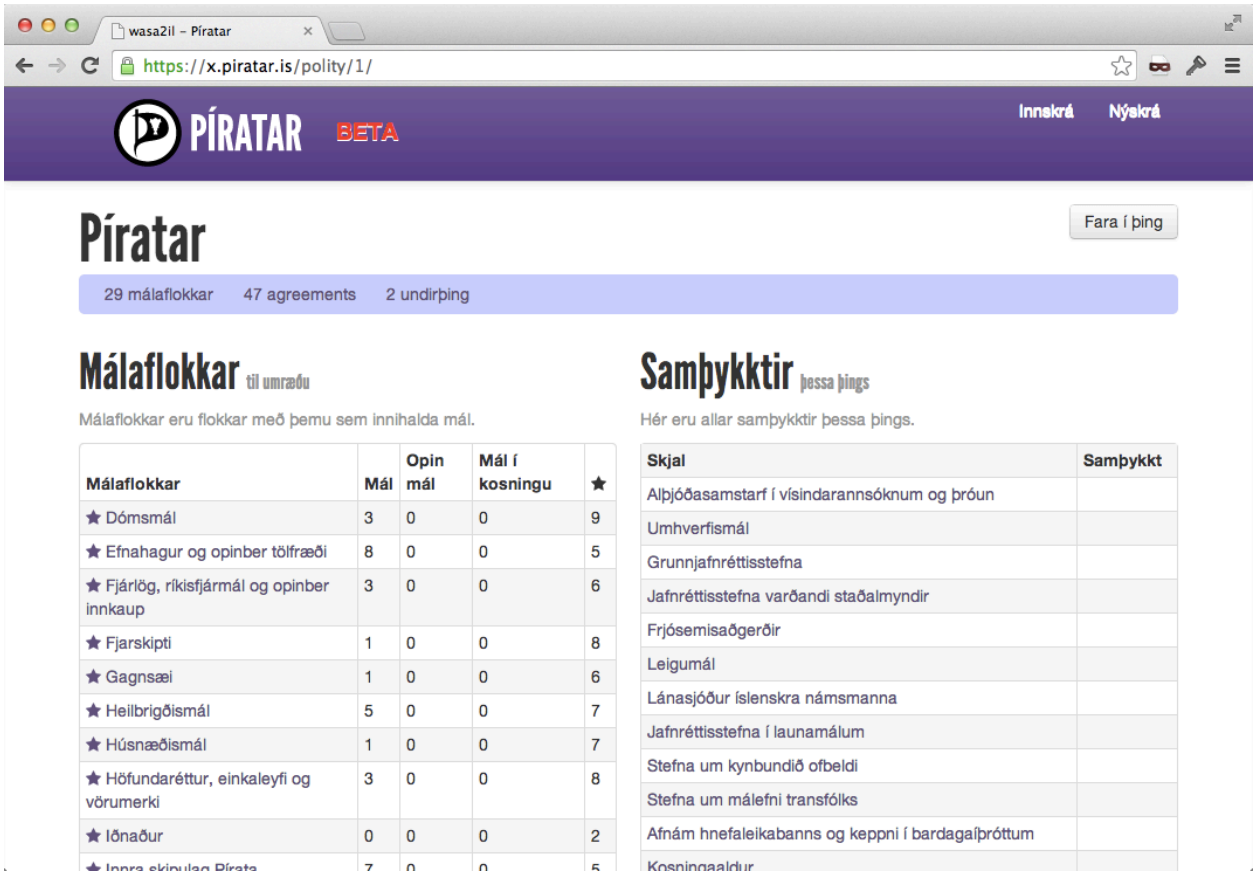
4.4 DemocracyOS

Name	DemocracyOS
Website	http://democracyos.org/
Code	https://github.com/DemocracyOS
Licensing	MIT
Programming Framework	Node.js, MongoDB
Self-Hosting	Yes
Test Suite	No test coverage

Developer Community	12 contributors; 5 significant, 1,400+ commits, weekly updates
Standards Compliance	No public API
Security, Anonymity, and Privacy	Standard web practice.

Very similar to Your Priorities, but redone in Node.js rather than Ruby on Rails. Needs more work on the test-suite and has less real-world usage than Your Priorities.

4.5 Wasa2il



Name	Wasa2il
Website	
Code	https://github.com/smari/wasa2il
Licensing	GNU General Public License v3.0
Programming Framework	Python, Django
Self-Hosting	Yes
Test Suite	No
Developer Community	9 contributors; 4 with significant contributions, 360+ updates, monthly updates
Standards Compliance	No public API
Security, Anonymity, and Privacy	Using best practice for the Web.

Wasa2il is used primarily by the Icelandic Pirate Party and features polling, proxing, and many other features.

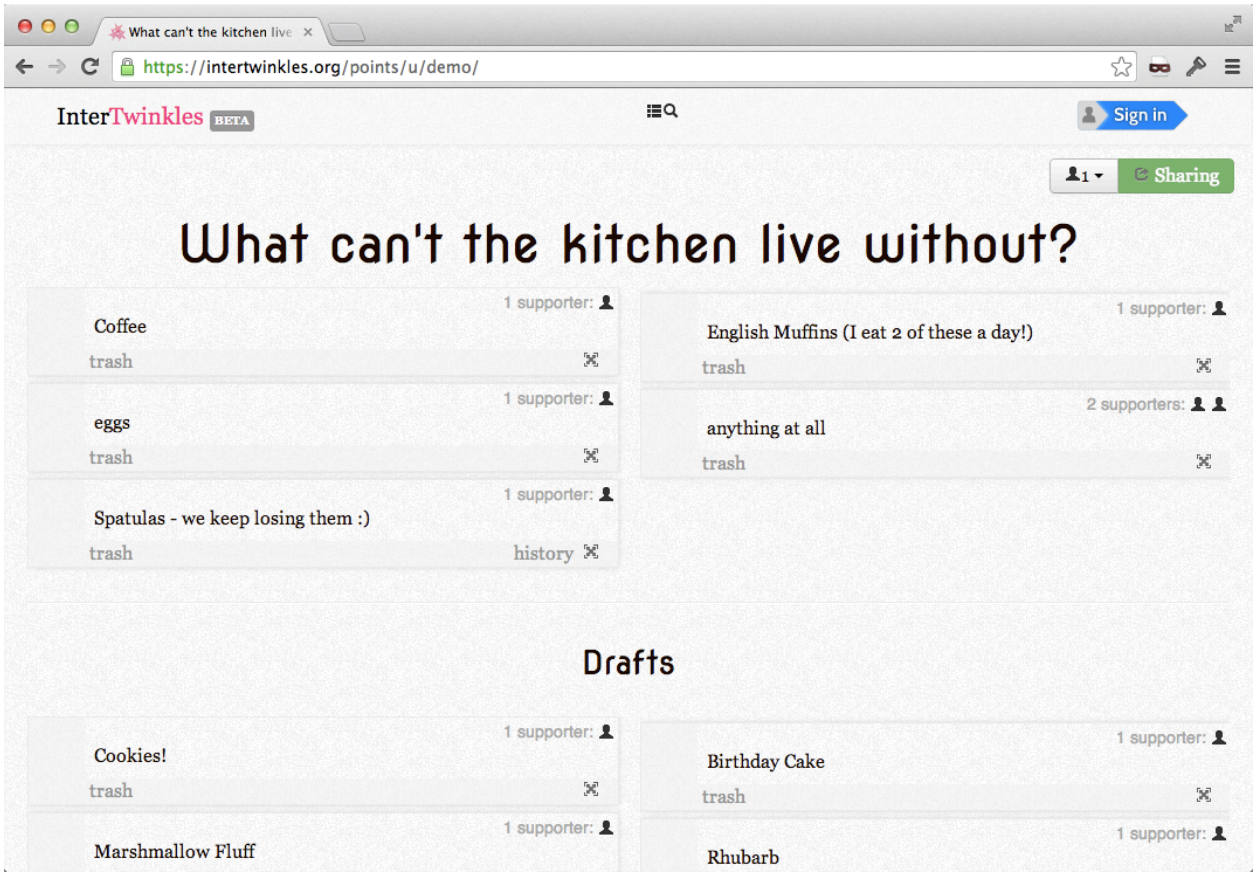
4.6 OpenMinistry

Name	Open Ministry
Website	http://www.avoinministerio.fi
Code	https://github.com/avoinministerio/avoinministerio
Licensing	MIT
Programming Framework	Ruby on Rails
Self-Hosting	Yes
Test Suite	Fair; partial coverage
Developer Community	Substantial; 18 contributors, 1,600+ commits, but inactive for last 9 months
Standards Compliance	No public API
Security, Anonymity, and	High authentication to prevent impersonation, using bank and mobile identities.

Privacy	However, currently not deployed due to some banks charging.

The codebase used by Finland to crowd-source new legislation. Very popular, but limited in functionality, except for the high value authentication.

4.7 Intertwinkles

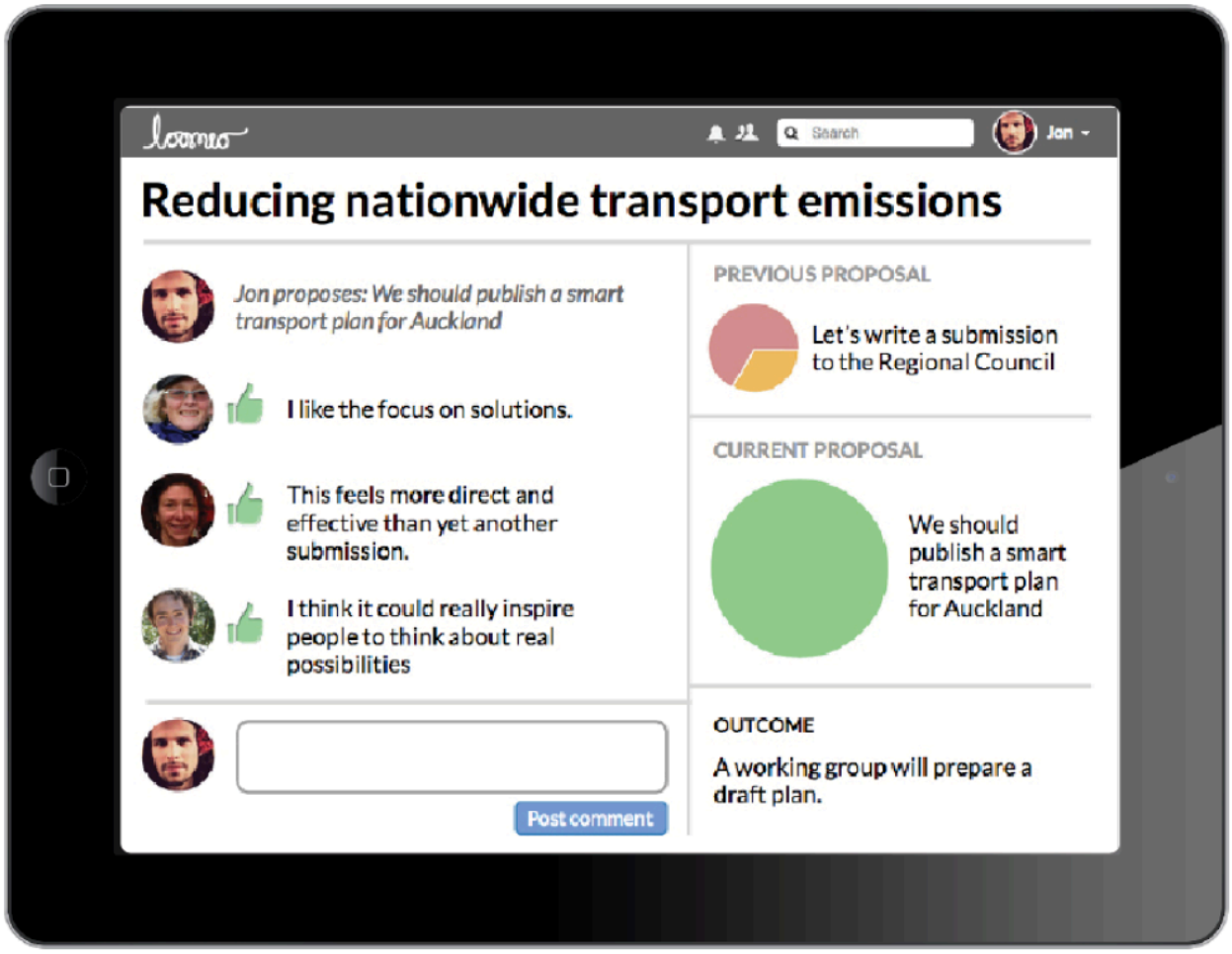


Name	InterTwinkles
Website	https://intertwinkles.org/
Code	https://github.com/yourcelf/intertwinkles
Licensing	BSD License
Programming	node.js, mongodb

Framework	
Self-Hosting	Yes
Test Suite	Good coverage, continuous integration
Developer Community	Minimal: one primary developer, one occasional contributor, 400+ commits, inactive for last 3 months
Standards Compliance	Minimal
Security, Anonymity, and Privacy	Ordinary best practices for the Web

The work of Ph.D. student at MIT, this code does feature a very good test-suite and simple features.

4.8 Loomio



Name	Loomio
Website	https://www.loomio.org/
Code	https://github.com/loomio/loomio
Licensing	GNU AGPL v3
Programming Framework	Ruby on Rails
Self-Hosting	Yes
Test Suite	Good
Developer	Strong, funded by commercial product, 4,500+ commits, 32 commits, daily

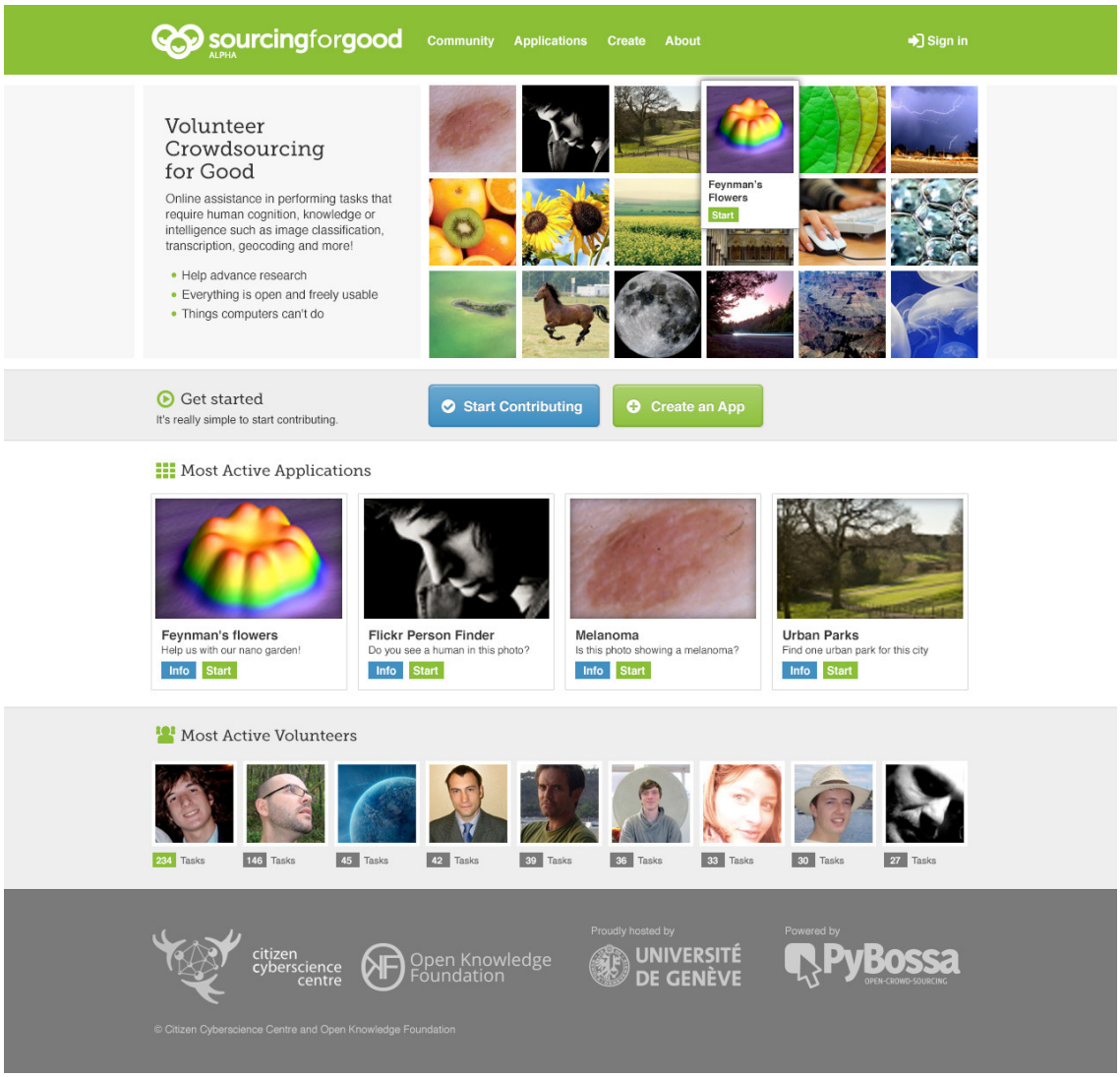
Community	updates
Standards Compliance	No public API
Security, Anonymity, and Privacy	Best Web practices.

An active codebase with deliberation and polling features, straightforward interface.

5 Open Data and Crowdsourcing Codebases

Since there are a vast amount of possible ways to use open data (APIs, XML, RDF) and a vast amount of work in the space, we will cover only codebases explicitly already used by partners are mentioned in the proposal. These proposals should allow us to meet open data integration technical requirement. Generally, these codebases feature some of their own social networking and limited polling capacities, which would be refactored in D-CENT.

5.1 PyBossa

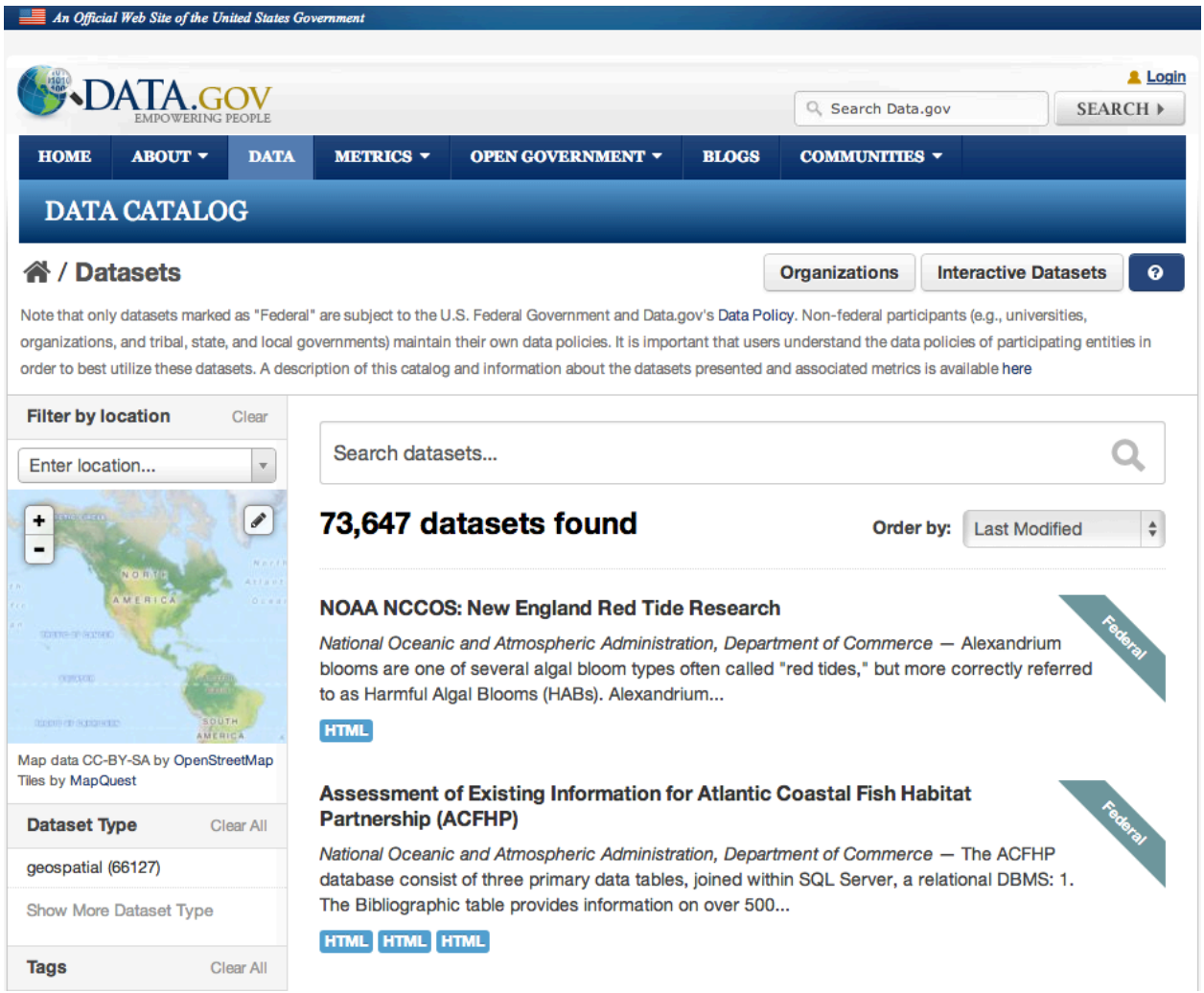


Name	PyBossa
Website	http://pybossa.com/ (http://crowdcrafting.org/)

Code	https://github.com/PyBossa
Licensing	AGPL 3.0
Programming Framework	Python Django with Redis
Self-Hosting	Yes
Test Suite	Started
Developer Community	17 contributors, 2,000 commits, daily updates
Standards Compliance	Standard
Security, Anonymity, and Privacy	No special considerations.

PyBossa is a open-source framework for managing micro-tasks in a similar way to Amazon Mechanical Turk, used by OFKN in the crowdcrafting.org site. However, it is not mostly developed by OKFN but actually by Universite de Geneva. Nonetheless, it seems to be a good option for crowd-sourcing if the pilots need that functionality.

5.2 CKAN



Name	*Comprehensive Knowledge Archive Network*
Website	http://ckan.org/
Code	https://github.com/ckan/ckan
Licensing	AGPL 3.0
Programming Framework	Python
Self-Hosting	Yes
Test Suite	Decent

Developer Community	10,000+ commits, 50 developers, daily activity
Standards Compliance	RDF
Security, Anonymity, and Privacy	Nothing beyond normal.

CKAN is an application to serve as a "data management system." However, it is not actually a database or datastore per se, but instead a metadata catalog of links to datasets, providing functionality to connect , categorize and preview open data. Access to metadata in RDF and JSON is provided through standard APIs.

5.3 CitySDK

Name	CitySDK
Website	http://www.citysdk.eu/
Code	https://github.com/waagsociety/citysdk
Licensing	MIT
Programming Framework	Ruby and Javascript
Self-Hosting	Yes
Test Suite	Little
Developer Community	6 contributors, 300 commits, monthly updates
Standards	RDF and JSON

Compliance	
Security, Anonymity, and Privacy	Standard web considerations.

The CitySDK framework is a lightweight "shim" from open data-bases to JSON and RDF, developed in the context of the EC-funded CitySDK project. Its shims have been used to expose data necessary for popular applications such as FixMyStreet. Although the codebase is not particularly active, given that it works and is already deployed in Helsinki and Barcelona, D-CENT will likely build from the APIs and data exposed by CitySDK.

5.4 OpenAhjo

Name	OpenAhjo
Website	http://dev.hel.fi/apis/openahjo
Code	https://github.com/City-of-Helsinki/openahjo
Licensing	European Union Public License v1.1
Programming Framework	Python Django
Self-Hosting	Yes
Test Suite	No
Developer Community	3 contributors, 270 commits, monthly updates
Standards Compliance	XML and JSON
Security, Anonymity, and	Standard web considerations

Privacy	

OpenAhjo is a thin shim between traditional databases used by the City of Helsinki and open data formats, with a focus on the decision-making information and records of the Helsinki. OpenAhjo covers the kinds of political data that is in general not covered by CitySDK. This kind of data will be necessary to be used by D-CENT, although the non-Finnish pilots will not use OpenAhjo.

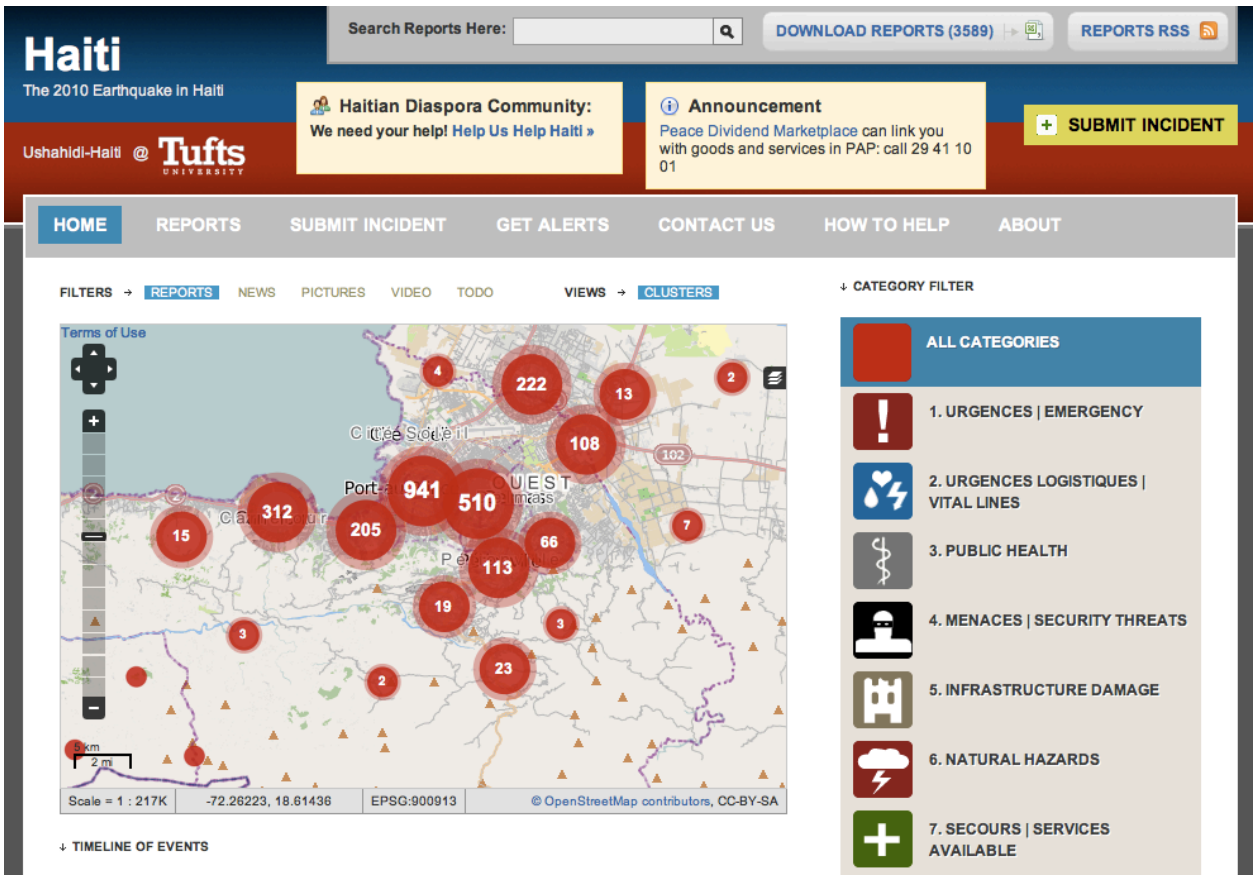
5.5 OpenStreetMaps

Name	OpenStreetMap
Website	http://www.openstreetmap.org
Code	https://github.com/openstreetmap

Licensing	GPL v2.0
Programming Framework	Ruby on Rails with PostgreSQL
Self-Hosting	Yes
Test Suite	Good
Developer Community	67 developers, 6,00 commits, weekly updates
Standards Compliance	KML (mapping standard), XML
Security, Anonymity, and Privacy	No special considerations

OpenStreetMaps is an open-source alternative to Google Maps and presents an open API for map data. It allows the publication of customized schemas and map visualizations. It also features a huge community of users, and accurate maps of most of the earth. In some areas such as Sarajevo, it is even more accurate than Google Maps (http://www.theguardian.com/news/datablog/2012/mar/28/openstreetmap-google-maps-technologies). For any mapping application, D-CENT should use OpenStreetMaps.

5.6 Ushahidi



Name	Ushahidi
Website	http://www.ushahidi.com/
Code	https://github.com/ushahidi/Ushahidi_Web
Licensing	GNU Lesser GPL v 3.0
Programming Framework	PHP and MySQL
Self-Hosting	Yes
Test Suite	Uneven
Developer Community	50 contributors, 4,000 commits, daily updates

Standards Compliance	KML (mapping standard)
Security, Anonymity, and Privacy	Usual problems with PHP and surprisingly little security or anonymity considerations given its use by human rights activists.

Ushahidi is a very well-known open-source mapping platform with innovative visualization and SMS features. Although originally built around Google Maps, it lets user's use OpenStreetMaps (although with a dependency on Google Maps for geolocation). However impressive its beginning, the community and codebase seem to be in a state of stasis, mostly due to the poor choice of PHP as a programming language framework. While features may be copied from Ushahidi, it is unlikely that D-CENT will build from the codebase.

5.7 Shardcache

Name	Shardcache
Website	Core library README https://github.com/xant/libshardcache
Code	Daemon implementation https://github.com/xant/shardcached
Licensing	GPLv3
Programming Framework	C++, SSL and a moderate range of Boost:: primitives
Self-Hosting	Yes
Developer Community	Shardcache is written by Xant. He uses it for work, yet he is proud to release it the GNU way. Jaromil can intercede with him. Xant is a talented perfectionist. This is likely to stay an artisanal piece, a'la Redis.
Standards Compliance	The style of code is very much that one of BSD C. The code is very portable, there are bindings to perl, python and go.

Security, Anonymity, and Privacy	Shardcache is a UNIX style daemon serving contents over ports, typically 80(http) or 443(https). It can be already used to distribute images to web browsers, efficiently distributing the load when serving many users.

Shardcached is a distributed cache and storage system for digital distribution of files and assets over http. It can keep up with a geographically diversified high-demand of contents by caching and serving files across a distributed network of servers.

6 Digital Currencies Codebases

Digital currencies such as Bitcoin are currently the centre of much attention. Adapting these codebases technically to community currencies in order to determine how they interoperate requires open-source code and a very flexible codebase. A few of the open source Bitcoin-style distributed ledgers used by cryptocurrencies are inspected here. Since these codebases are very new and not in general developer-facing, there are no test-suites yet.

6.1 BitcoinD

Name	BitcoinD
Website	http://bitcoin.org
Code	http://github.com/bitcoin/bitcoin
Licensing	GPL2
Programming Framework	C++, SSL and a moderate range of Boost:: primitives
Self-Hosting	Yes
Developer Community	The community of developers around BitcoinD is very large and competitive. Its current maintainer is in charge since almost 4 years now and a foundation based in USA is trying to gain approval to represent the community.

Standards Compliance	Stratum (protocol), Bitcoin blockchain, currently approved BIPs Protocol specification: https://en.bitcoin.it/wiki/Protocol_specification BIP Index: https://en.bitcoin.it/wiki/Bitcoin_Improvement_Proposals
Security, Anonymity, and Privacy	The Bitcoin blockchain collects an history of pseudonymic transactions, which are basically contracts. Anonymity can be granted by masking network access points for transactions.

BitcoinD is the first reference implementation for the Bitcoin mainnet blockchain of transactions, originally authored by Satoshi Nakamoto and nowadays maintained by a large community of developers and stakeholders. Maintainance of this codebase is fairly conservative and responds to a series of RFC-like improvement proposals called BIPs. Most Bitcoin derivatives today have forked from this codebase, customizing only certain parts, most often the genesis block, the proof-of-work algorithm and the p2p distribution of transactions.

6.2 Libbitcoin

Name	Libbitcoin, SX (CLI), Obelisk (Daemon)
Website	http://libbitcoin.dyne.org
Code	https://github.com/spesmilo/libbitcoin
Licensing	AGPL3
Programming Framework	Modern C++ standards (C++0x) and Boost:: primitives
Self-Hosting	Yes
Developer Community	A big part of the Bitcoin community from the “golden days”, also many young and talented programmers being attracted by the popularity and trying to submit code into the reference implementation, which got mostly rejected. Naturally some started dreaming of a rewrite. Libbitcoin's rewrite takes advantage of C++0x and newer Boost primitives. The project has a well planned architecture and a charismatic talented leader.

Standards Compliance	Stratum (protocol), Bitcoin blockchain, currently approved BIPs ad-hoc: https://wiki.unsystem.net/index.php/Libbitcoin/Blockchain
Security, Anonymity, and Privacy	Libbitcoin inherits all Bitcoin's features regarding security, anonymity and privacy of its operations.

Libbitcoin is the ground-up rewrite of the Bitcoind codebase, after about 2 years of study and empirical experimentation, fully compatible with the BTC blockchain. It provides python bindings and uses a client-server protocol to let lighter client side implementations refer to an on-line blockchain whose blocks are served on demand by a daemon.

6.3 Picocoin

Name	Picocoin
Website	https://github.com/jgarzik/picocoin
Code	
Licensing	GPL3
Programming Framework	Very portable POSIX C
Self-Hosting	Yes
Developer Community	One main talented developer with experience of Linux kernel coding, plus a small but growing group of contributors and fans.
Standards Compliance	Bitcoin blockchain, currently approved BIPs
Security, Anonymity, and	Picocoin inherits all Bitcoin's features regarding security, anonymity and privacy of its operations.

Privacy	

Picocoin is the ground-up reimplementation of Bitcoind, starting from a blockchain explorer, in simple and clearly written C language. It is fit to run on embedded setup, requiring little overhead.

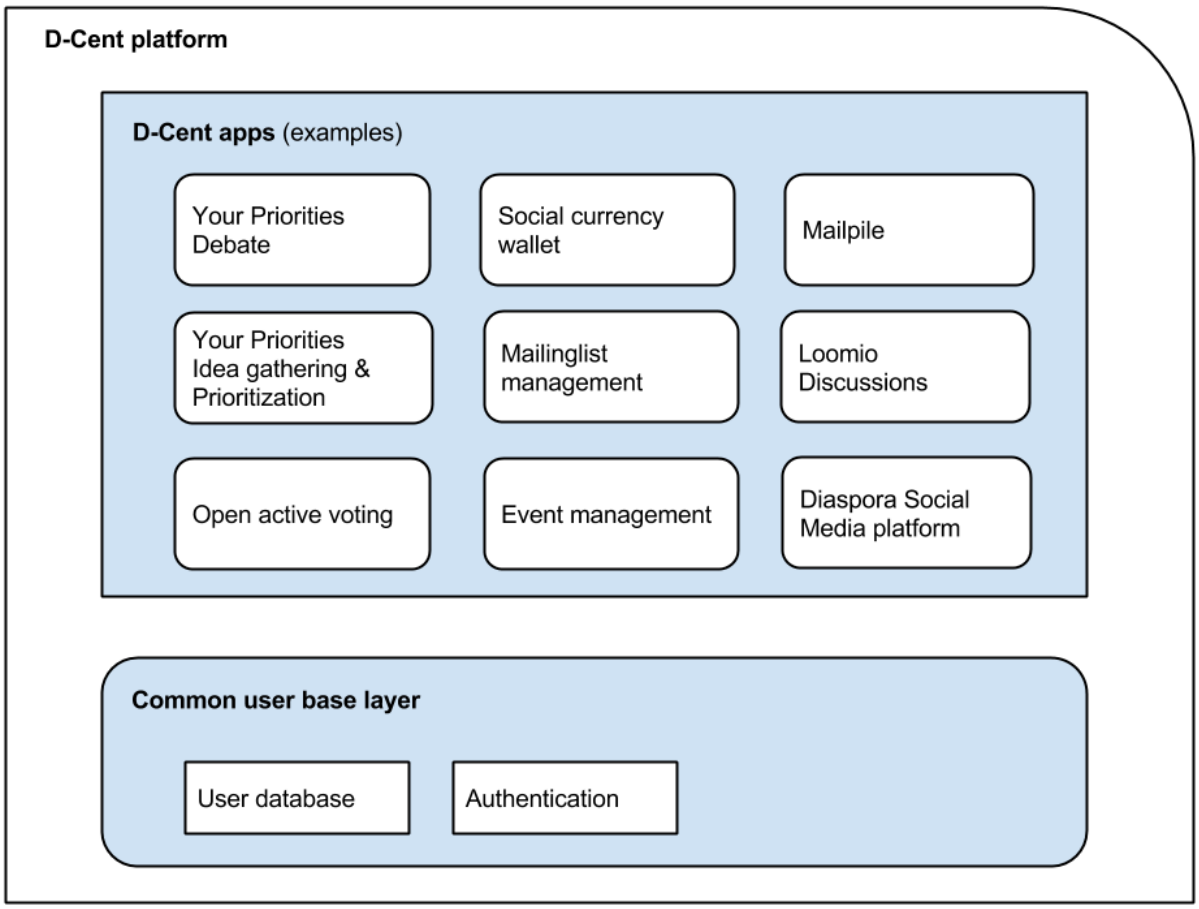
7 Initial D-CENT Architecture Design

7.1 D-CENT Platform

The general D-CENT platform should, using the lean methodology, build from the lightweight pilot MVPs detailed technically below. Once the first version of a MVP reaches the technical requirements given below, the primary components that can be shared should be factored out with the shared code beneath being tested using the lean methodology.

The core concept for the D-CENT platform is to create a set of standard based APIs that enables D-CENT application pilots to share the same user base on a given D-Cent node as well as commonly used functionality. This prevents pilots from "reinventing" the wheel.

Unfortunately, we cannot chose a single codebase that provides all of this core functionality "out of the box" because the current landscape of social networking codebases is fragmented, divided between an odd-mismash of Facebook and Twitter clones, with many code-bases being abandoned or having only a very small development community and no clear codebase meeting all the social and technical requirements, as detailed in the review of the codebases. Thus, instead we will use standard Ruby and Node.js libraries as exist from Diaspora and Pump.io respectively to build in the course of the MVPs the necessary social networking components. The results of these programming work will be given in D5.1.



We should fully expect that the authentication, authorization, personal/social data, and status update messaging should be factored out of each MVP eventually, replacing them with standardized interfaces to standards-based libraries. As justified in D4.1, for authentication we will always provide a standard email and password login, but allow authentication using Facebook Connect and Twitter as needed. Secure Remote Password and the use of the W3C Web Cryptography API will be explored, along with national-level eID schemes if available for free, when needed. For authorization, OAuth2.0 will be used, and for messaging, ActivityStreams 2.0 will be used. Based on best practices, for the user-databases, we expect to use PostgreSQL (http://www.postgresql.org/) for structured data that requires schemas and Redis (http://redis.io/) for noSQL attribute-value pairs. We expect a more detailed architectural document for the social networking and standards to be evolved by D4.3 as the pilot experiments continue.

The large architectural choice is between an XMPP-based architecture (exemplified by BuddyCloud) and a pure HTTP architecture (as exemplified by Pump.io). Following a lean approach based on modern Web development, dragging in the entire complex XMPP stack at an early stage would probably be a mistake. It would make more sense to attempt to build with pure HTTP first and then slowly move to standards-based on HTTP, and only moving to XMPP-based protocols if absolutely necessary, and then in a step-wise manner, adding only minimalistic XMPP features on an as needed basis. Since D-CENT will likely not build on top of Kune due to the dated and heavyweight nature of Wave codebase and its XMPP dependencies, bridges between P2PValue's Kune deployment and the D-CENT platform should be established. The second problem is what codebases to build from. Diaspora is a Facebook clone, and is too heavy-weight for many MVPs to be introduced all at once.

However, components from Diaspora could be introduced again, as needed, in a piece-meal fashion. For high-volume processing, although the codebase has gone fairly inactive, the Node.js framework Pump.io would be suitable. Although the lack of use of Rails and the dated nature of the code require re-factoring, for some group-based access control, the Crabgrass codebase may have some re-usable components. As stated earlier, the social networking component should move in small steps rather than wholesale adopting a particular code-base.

7.2 Licensing

D-CENT likely will have to use code that is not licensed by the GPL, as some codebases use MIT and Apache licensing. However, new code will use GPL v3.0 for client and AGPL v3.0 for new code. Copyright will always be maintained by authors, except in the case of Neo, in which copyright will be given to the W3C as the W3C has agreed to host the software in perpetuity on W3C's DCVS system as well as copy toGitHub

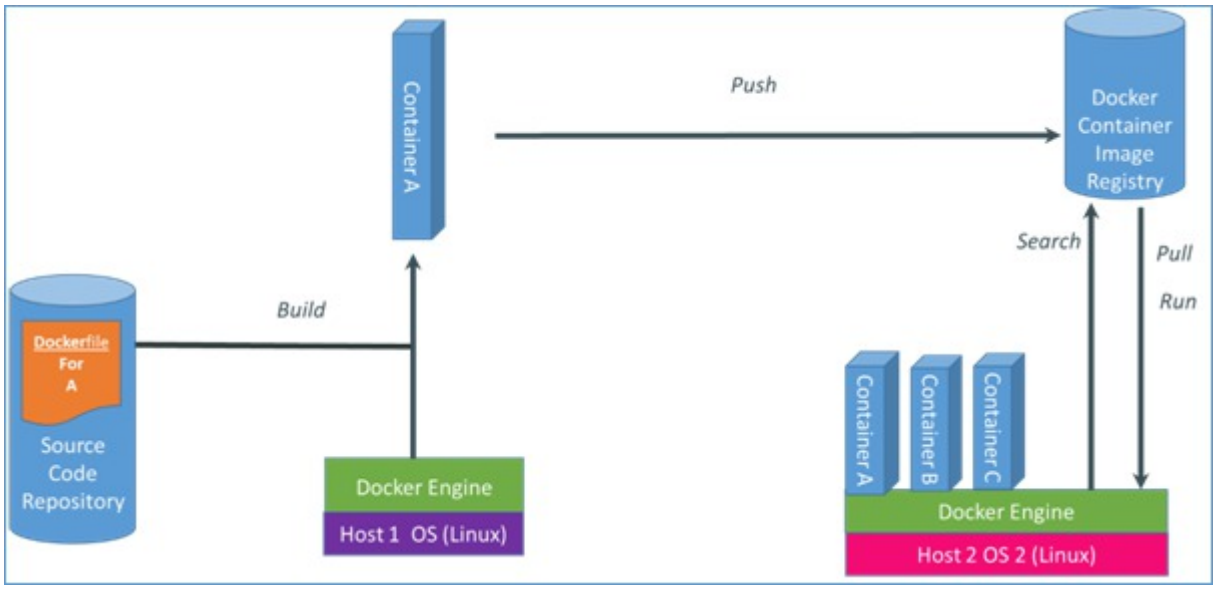
7.3 Programming Framework

The D-CENT platform should make the best usage of modern Web application development. Although any component may be interfaced with using standard JSON-based APIs, in "lean" development for new applications a mixture of Ruby on Rails and Node.js is often preferred. In particular, Ruby on Rails provides a large amount of powerful libraries and a great abstraction interface. However, this comes at the cost of overhead. Thus, for optimization purposes, high-usage aspects may require many concurrent calls should use pure Javascript and Node.js. In general for D-CENT, most useful components are built on top of Python, Node.Js, and Ruby, although the digital currency codebases are mostly in C. Luckily, using Docker, the D-CENT platforms welcomes applications written in any language of choice including Ruby on Rails, Node.js, Python, C++ and any other language that will work on Linux.

7.4 Self-Hosting

The D-CENT platform and each MVP will mature in a way such that it can be self-hosted. The requirements gathered in D1.2 tell us that the D-Cent platform needs to be able to encompass apps written in many different programming languages. The solution that D-CENT has adopted for early work in self-hosting is Docker (http://www.docker.io), which is

"an open-source engine that automates the deployment of any application as a lightweight, portable, self-sufficient container that will run virtually anywhere." In particular, it should run on OpenStack, virtual servers, and complete self-hosted environments, and thus covers most of the prospected needs of D-CENT. Docker makes it easy to build, modify, publish, search, and run containers. The diagram below should give you a good sense of the Docker basics. With Docker, a container comprises both an application and all of its dependencies. Containers can either be created manually or, if a source code repository contains a DockerFile, automatically."



D-CENT platform

7.5 Developer Community

Since the D-CENT work has just begun, we do not yet have a large developer community. However, by building off Ruby on Rails and Node.js frameworks as well as libraries from popular components, such as Diaspora and Pump.io, we expect that attracting a large developer community will be possible. In the project, there is already a large user community around Your Priorities and a community of programmers who work with Forum Virium, including Code For Europe fellows, interested in open data and democracy in Helsinki.

7.6 Standards Compliance

Surprisingly, support for standards (see D4.1) is generally poor amongst all codebases. From the perspective of social networking, there is some support of the OStatus stack, but the stack itself is rather dated and in need of overhauling or rewriting in the W3C, which is likely to be influenced by the more lightweight IndieWeb work. However, for social messaging there is strong support in general of ActivityStreams, although no codebases yet support the ActivityStreams 2.0 standard that would work with Linked Data. This is important, as many of the APIs such as CitySDK use Linked Data and JSON, and so modifications are likely to be made to Pump.io's JSON libraries to upgrade them to the new version of ActivityStreams. Also, almost all of the social-networking and digital democracy code-bases have very poor authentication and authorization components, and very little in the way of actual personal and social data portability via vCard. Luckily, open-source Ruby libraries such as OmniAuth for authorization (http://intridea.github.io/omniauth/) and authentication exist for most of this that can easily be added to the D-CENT platform, with later developments around the W3C Crypto API and national-level eID schemes being exploited as they mature. Diaspora features hCard support for exporting vCards for personal data, but some work may need to be done in-order to fully modernize it with the latest versions of vCard and exporting social graphs. One open

question is how the social graph itself should be structured in the D-CENT platform, but a social graph server using a graph-based database such as Neo4J (http://www.neo4j.org/) may be route if appropriate vCard modifications for export and import of personal data do not easily fit within the Redis/PostgreSQL stack. In general, we expect the standardization component of the D-CENT platform to be developed in tandem with the W3C Social Web Working Group (http://www.w3.org/2013/socialweb/social-wg-charter.html).

7.7 Security (Privacy and Anonymity)

In general, security was not taken seriously by most codebases with the exception of TextSecure and LEAP. Only OpenMinistry provided high-value authentication using national eID standards. Outside of these, no codebases did anything above average to defend usernames and passwords, and almost no codebases used digital signature to verify data or any form of encryption beyond standard TLS and XMPP practice. The exceptions would be LEAP, which together with Mailpile should provide full client-encrypted email, and TextSecure, which provides secure messaging – which are both non-Web applications. Web-based frameworks such as Node.js and Ruby on Rails are by nature controlled by the Web server and thus in general not able to provide secure client-encrypted storage. In particular, although TextSecure has made some progress, even providing group-based messaging in an encrypted and secure manner (multi-party OTR) has also not been solved even inside non-Web based applications. This group-based messaging with some form of access control of course is necessary for social networking and digital democracy. Thus, in terms of security and privacy the longer-term goal for D-CENT should be to adapt ideas from TextSecure to social networking when they mature and as the Web itself matures. In the meantime, TextSecure and Leap/Mailpile should be used for text-messaging and Mailpile when called for by pilots like the Spanish pilots. For Web-based applications, the use of Secure Remote Password (http://srp.stanford.edu) should be experimented with in order to strengthen the database of username-passwords for the D-CENT platform. Once the D-CENT platform and applications are mature (D5.3 and D5.4), a thorough penetration testing should be done to avoid common security errors.

Privacy was in general respected across most reviewed code-bases, all applications in general did not force the user to disclose unnecessary information, and with the exception of the use of nation-level eID in OpenMinistries, codebases did allow the use of pseudonyms. Users could control their user-data. Most programs did not allow OAuth flows for authorization of personal data change with the exception of Diaspora. However, users were not made aware of any authorizations made by OAuth in Diaspora using standards such as User-Managed Access Control (See D4.1 for a description. Future work on D-CENT should make clear terms-of-service and consent for the movement of user-data. User data could in general not be portable except via the use of hCard/VCard in Diaspora, Pump.io, Status.Net, and Elgg. D-CENT should continue these good practices and re-use their code libraries when needed.

Almost no codebase takes into account anonymous usage. In particular, anonymity techniques such as message padding before encryption or cover traffic have not been used, or even zero-knowledge proofs for identity authentication. Due to the general lack of work in privacy and anonymity and security in these codebases, it would not make sense to test analytically the privacy, anonymity, or scalability of such work at this stage. If needed, such tests should be done after the completion of

D4.3 if questions arise about the suitability of any security, privacy, or anonymity techniques and whether or not they will scale.

This lack of work on security and anonymity, and still fairly immature work on privacy, in federated and decentralized social networking systems is a major research gap. It cannot be addressed in the D-CENT project easily due to the lack of privacy, security, and anonymity researchers in the project, but should be addressed in research projects after D-CENT. The W3C should be able to enable standards-based solutions that, while not cutting-edge, do at least guarantee the users of D-CENT better security, privacy, and anonymity in D-CENT than in commercial providers like Facebook and Twitter.

8 Conclusion: Translating the Technical Requirements into Pilots

As D-CENT is following a lean approach and no single suitable codebase could be found, the lean approach will develop three pilots that then, over time and in D4.3 and D5.1, mature into the D-CENT platform. The technical requirements that come from the social requirements should then be tested in the pilots. This will determine what parts of various codebases reviewed here are useful for the future of the D-CENT project as well as whether or not the technical requirements adequately represent the social requirements as discovered in D1.2. As we are working with the" Lean" method we have selected a couple of Minimum Viable Products to test in each of the pilot countries. Below are technical requirements for the first lean Active Experiments in all three pilot countries.

8.1 Iceland Pilot

There are two initial MVPs that will be tested in Iceland. The first will be "rating" town hall responses to citizen proposals to Your Priorities. This will technically result in adding a rating system for responses from town hall and a measurement system to measure the impact of the feature on participation and the quality of responses back from the local government.It should be tested over a number of iterations of feedback from city government.

Diaspora, which is a Ruby on Rails application, will be launched in Iceland using a Docker container to see if it can help with deliberation. Diaspora needs to be deployable in a reliable way with high availability and will be made available as an option for connecting to friends in Better Reykjavik and Better Iceland that are running on Your Priorities. D-CENT needs to build on an integration with Your Priorities to identify a users' friends on Disapora and share content with them. In order to drive more participation, there will also need to be a feature to automatically create an user if people are coming in from already accepted Facebook account. The same Facebook API key will be used for both Diaspora and Your Priorities.

8.2 Spain Pilot

The Spanish Pilot MVP will focus on adding notifications that better fit into to workflow of activists who need to make quick decision-making. This can initially be done with simple Ruby on Rails development, but later we expect that the status updates archiving and customization should be added in an additional MVP that builds from the initial one, as well as group-based deliberation over decisions that need to be done. Just as in Iceland, the customization and use of status updates will likely lead to re-use of some of Diaspora and Pump.io's underlying libraries. An additional Spanish MVP is currently engaged with tracking the user metrics of ComPAHS. However, as that software is not open source, it would make sense at some point, if the MVP shows a high amount of use, to refactor that codebase in order to create an actual secure messaging component. For chat, that would likely mean switching to TextSecure, and for email that would mean switching to some combination of LEAP and Mailpile, with OpenStreetMaps for map-based data integration. Task management is still a major missing component and some software may have to be built to address this technical requirement. The decision-making component can build from whatever software is produced by the first Spanish MVP. All components should end up using ActivityStreams standards in order to produce self-hosted archived and customized status updates.

8.3 Finland Pilot

The Finnish Pilot will require the most intensive open data integration, combining that work with improved notifications. In particular, there will be lean experiments to see if we can have notifications of townhall meeting agendas. However, like in Spain, the structuring of the workflow in terms of the technical requirement of task management has no open-source software that clearly does the work, and this software may have to be developed. However, in terms of experiments, Finland may wish to attempt to run a MVP using DemocracyOS with additional underlying libraries from pump.io to address the technical requirements around polling and proxying, in addition to inspecting the use of the Your Priorities codebase. One of the harder problems facing Finland (and Iceland, if proxying experiments go well) is the lack of high-value standard authentication, as obviously without this feature any voting could be compromised. However, currently OpenMinistry had to disable this feature due to banks charging up to one euro per authentication for this feature, so it will not be included in any initial pilot work.

8.4 Digital Currency Pilots

The technical requirements of the digital currency pilots are aimed at the hard problem of digital currency exchange. The digital currency pilots are still very much in the early stages, with the main concept connecting them to the D-CENT platform being to discover a way to share the same distributed blockchain. We expect this to be detailed more in D4.3 and D4.4. At this early stage, since the digital currency pilots may need more heavy-weight client software than the rest of the pilots, the codebases to be used include GNUNet and the bitcoin codebase being Libbitcoin.

8.5 Conclusions

Although ambitious, the D-CENT pilots are clearly addressing needs not addressed in current centralized social networking platforms such as Facebook and Google. No single open source codebase can fix all of these problems, but instead more engagement with the community and work with the pilots need to be done. However, the community has already expressed many social needs that can be translated into technical work. In particular, the need for security and privacy is crucial, as well as the need for better integration into existing open data produced by the community. Likewise, there is a tension as communities both want to own their own data yet not lose the "network effect" that comes from communicating via their normal activity streams. Overall, current workflows used by citizens throughout Europe are too fragmented across far too many communication channels (mailing lists, Twitter, Facebook) and lack the structure and data protection needed to accomplish real work. We believe that, following the lean approach, D-CENT can commit experiments that show what kinds of tools people need to structure their workflow and attention, and so lead to a new generation of decentralized applications for collective awareness.