


# INTERNATIONAL UNIVERSITY

## Pre-Thesis

### Decentralized Online Social Networking Analysis Report

▼ Dan Brickley



☐ This is you


**Basic Information**

<a href="#">danbri homepage</a>
<a href="#">my.opera homepage</a>
<a href="#">danbri.livejournal weblog</a>
<a href="#">my.opera weblog</a>

Openid ► <http://danbri.livejournal.com/>

Nickname danbri  
danbri

**Acquaintances**

Aaron Swartz
Amy van der Hiel
Art Barstow
<a href="#">Benjamin Joffe</a>
<a href="#">Charles McMathieNevile</a>
Damian Steer
<a href="#">Dan Connolly</a>
Dave Beckett
<a href="#">Dean Jackson</a>
<a href="#">Edd Dumbill</a>
<a href="#">Eddie Lopez</a>
<a href="#">Eric Miller</a>
<a href="#">Eva Méndez</a>
<a href="#">Gregory J. Rosmaita</a>
 Jan Grant
Jim Ley
Joe Brickley

Tip: Do you have [web ID](#)?

[Make or set A Web ID](#)

SeeAlso ► <http://foaf.qdos.com/reverse/?path=http://danbri.org/foaf.rdf#danbri>

**Advisor:** MSc. Vo Duy Khoi  
Dr. Tran Manh Ha

**Student:** Le Quoc Thanh  
**ID:** ITIU09028

## Contents

<b>I. Overview.....</b>	<b>3</b>
<b>II. Decentralized Online Social Networks.....</b>	<b>4</b>
1. Introduction: .....	4
2. Challenges for Decentralized Online Social Networks .....	5
3. The Case for Decentralized Online Social Networks .....	6
4. General purpose Decentralized Online Social Networks.....	7
4.1 Proposed Decentralized Online Social Networks approaches.....	8
5. Specialize Application Centric Decentralized Online Social Networks.....	10
5.1 Social-based P2P File Sharing.....	10
5.2 Shared bookmarks and collaborative search.....	10
5.3 Micro-blogging .....	10
6. Social distributed systems: .....	11
6.1 Social DHT: Social Circle:.....	11
6.2 Storage / Back-up:.....	11
7. Delay-Tolerant Decentralized Online Social Networks: .....	11
<b>III. Architectures for Decentralized Online Social Networks .....</b>	<b>11</b>
1. PrPl: .....	11
1.1) Introduction.....	11
1.2 Federated id management.....	13
1.3 The PrPl semantic index.....	14
1.4 Socialite: A Language for a Social Multi-Database.....	15
1.5 Prototype Implementation.....	16
1.6 Application Experience.....	16
2. SuperNova: .....	17
2.1 Introduction.....	17
2.2 Approach.....	18
2.3 Evaluation.....	21
<b>IV. Conclusion.....</b>	<b>23</b>
<b>V. Reference.....</b>	<b>24</b>

## I. Overview

The current centralized of Online Social Networks has several drawbacks including scalability, privacy, dependence on a provider, need for being online for every transaction, and lack of locality.

Actually, for social networking's Web sites present problems as:

- These sites form information silos. Information on one site is not usable in the others.
- Such sites do not allow users much control over how their personal information is disseminated, which results in potential privacy problems.
- The companies providing the services have the sole authority to control all the data of the users.
- Users usually have little control over how and what information about them is presented to their friends online.
- Users have to agree to the policies of the social networking sites when using their services, even though they may involve usage of their data for targeted advertising.
- Very often users need to explicitly opt-out of certain applications if they are more conscious about the privacy of their data.
- Hinder creativity because of imposing restrictions on how new applications using the social graph can be created.

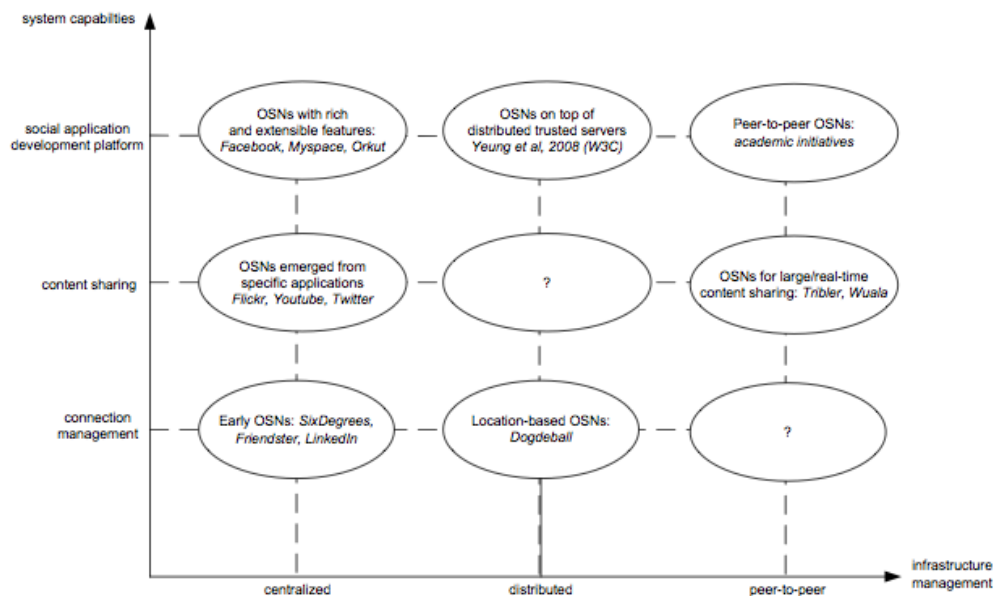
However, there is a proposal for Decentralized Online Social Networks (DOSNs), which can solve these problems by three respects:

- Privacy: Users in decentralized social networks decide who to show the information to and what restriction there is on the data.
- Ownership: As the information is stored on a trusted server or on the local computer, users have complete ownership of the data. They would not lose their data suddenly as the proprietary service hosting their data decides to shut down without much notice. [Wilson 2008]
- Dissemination: Information is disseminated according to users' preferences and friendship relations.

## II. Decentralized Online Social Networks

1. **Introduction:** Online Social Networks (OSNs) defined in 3 main parts:

- Providing services for user to build a public profile and to explicitly declare the connection between his/her profiles with those of the other users.
- User share information and content with the chosen users or public.
- Supporting the development and usage of social applications to help user can interact and collaborate with both friends and strangers.



**Fig. 1.** Classification and development trend of online social network services

The current online social networks are extended in two main directions towards the capabilities of the provided services and the decentralization of the supporting infrastructures.

Centralized OSNs services are prone to some problems including:

- Technical, which rapid growth in user popularity has led to various performance scalability issues increasing cost of management and maintaining the infrastructures to ensure a smooth continuation for services,
- Social side, which unlimited sharing capability of information has also led the social collisions and proper privacy preserving schemes, leads to a collapse in social contexts.

Therefore, DOSNs is a trend for current OSNs services, which implemented on a distributed information management platform as network of trusted servers or peer-to-peer systems. In addition, distributed or peer-to-peer

OSN offers a cost-effective alternative, better control of user privacy (no central data collection, reduced economic incentive for advertisement), and enhancement of innovative development.

## 2. Challenges for Decentralized Online Social Networks

Decentralizing the existing functionality of online social networks requires finding ways for

- Distributed storage of data.
- Updates propagation and versioning, a topology and protocol to search and addressing, a mechanism to find friends in the topology, robustness against churn, openness for third-party applications, and means for content revocation.

There are some challenges for decentralized social networks as:

- **Storage:** The requirement for redundancy to provide availability of data depends to a large extent on the duration and distributions of time peers are online. These activity patterns are also influenced by the geographic distribution of the peers and shifted by time zones
- **Updates:** in peer collaboration systems, updates a workplace are sent to a small group of peers via a decentralized synchronization mechanism. Unlike a traditional peer-to-peer environment, many peers are involved; each of the sub-networks will be much smaller making it relatively simpler to realize quorum systems and deal with updates.
- **Topology:** in the widest sense, mostly platforms for collaboration or media sharing and they tend to consist of collaborative groups that are relatively closed circles, using a “ring of trust” or dark nets. In contrast, online social networking services have overlapping circles.
- **Search, Addressing:** peers may change their physical address. In a typical file-sharing network, this is not an issue that it just needs to find some peer with the content it is looking for. In social networks, tagging or folksonomies is the basic mechanism to annotate content, which paves another step towards realizing social networks on top of a P2P infrastructure.
- **Openness to New Applications:** core functionality for maintaining social ties, such as profile information, connection to friends, status updates, internal messaging, posting on each other’s sites, events notification. In decentralized environment, if some users choose to enable a third-party application, their choice should not affect other users or even users connected directly to them
- **Security:** the classical requirements for security (confidentiality, access control, integrity, authentication, non-repudiation) apply, albeit modified for the context of decentralized social networks. For distributed storage with other peers that the user’s content has

to be encrypted and to manage this data that key distribution and maintenance have to be handled such that be flexible enough to handle churn in terms of going offline and coming back, additions and removal to the user's social network. There is on top of that the need for access control to determine who can read, write or modify and delete each shared object, while still guaranteeing non-repudiation as well as preventing impersonation and replay. Other security issues like prevention of DDoS and Sybil attacks cooperation and preventing free-riding or content pollution, and establishing trust are also of course long-standing issues in the community.

- **Robustness:** In a centralized system, one can turn to the provider in case of user misbehavior, which process defined for dealing with such complaints. In a decentralized system, there is no authority that can ban users for misbehavior or remove content.
- **Limited Peers:** mapping of physical social network to virtual and vice versa enables extensions to offering access via web browsers by phone applications and direct exchange of data in physical proximity.
- **Locality:** real-life social networks can be used to support the decentralized social networking application. Also, distributed architecture also enables us to take advantage of geographic proximity and its correlation with local interests

#### \* Differences to other decentralized or P2P applications:

Peer-to-peer storage has been done successfully for file-sharing which copy of a music, video, media file, potentially present in high numbers. However, they are usually not updated, although new versions or different content get added to the system.

In social networks, current status of a person is updated often and value of outdated information is much less than that of timely information that enables users to react to content changes.

For peer-to-peer social networks privacy is more important as it concerns personal information, so storing content unencrypted at other peers whom the user does not want to access personal information is not an option. Files should only be readable by peers that are specifically allowed to access them. Peer-to-peer collaborative work are added or removed from a working group at a lower rate than expected churn for social networks. For DOSN we need both in terms of online/offline behavior and of adding/removing friends and corresponding access rights. In addition, they also need a large-scale peer-to-peer network with fine-grained access control for reading and writing, with changing files (versions), file-sharing, chat, news-feeds, public and private asynchronous messaging, search, notifications

### 3. The Case for Decentralized Online Social Networks

The information gathered can be used for data mining, direct

advertising, censorship, or other purposes. Moreover, a centralized depository or fully connected network is more susceptible to virus or malware spreading than mostly local social networks that can be partitioned. An immediate advantage of DOSNs is rather straightforward: it is not centralized, not owned by a single entity.

In DOSNs, privacy has become a major concern. Particularly privacy and protection from massive data mining and “big-brotherly” treatment of the users by the social networking service providers.

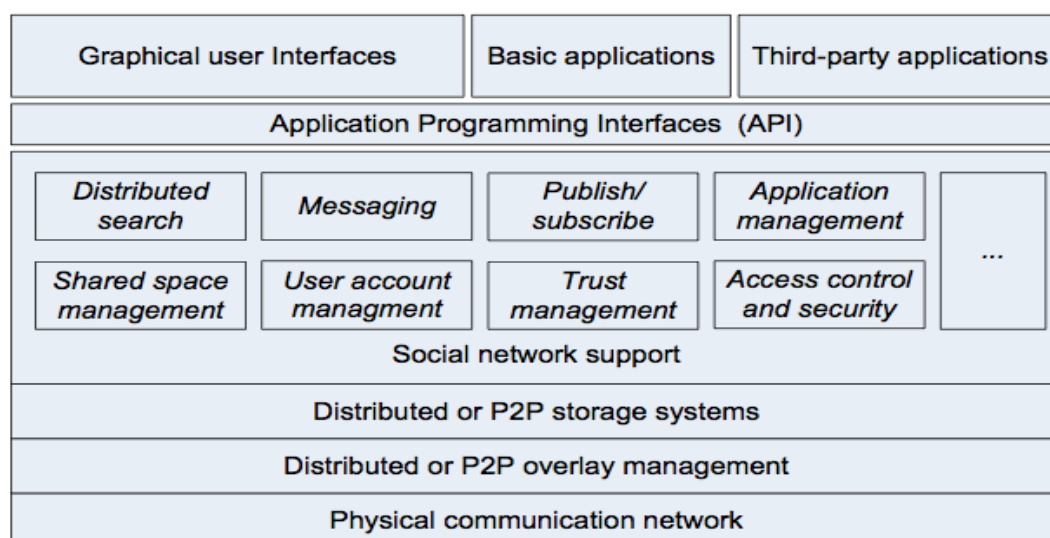
With a peer-to-peer approach, decentralization is given, and combined with appropriate encryption users can determine whom they allow access to their data. Another incentive for users Decentralized approach promises to be the right technology to achieve both privacy and freedom of speech better than a client-server model.

Also, peer-to-peer approach is in ensuring user control. First, whoever would be willing to provide the centralized service and infrastructure would also be able to cease to provide the service or change its terms. Second, due to the lack of data mining and advertising possibilities, provide a good service and all the servers necessary for a centralized solution. Third, a centralized service requires more trust by the users than a distributed system that limits the risk of privacy breaches by not providing a central repository of user data.

In addition, peer-to-peer social network that separates these spheres and enables users to maintain their social network without commercial prompting by advertisement. About user control of data that they can control over who can access their content and what they are allowed to do with that content and users also can enjoy freedom of speech, without fearing censorship or other obstacles.

#### 4. General purpose Decentralized Online Social Networks

The reference architecture consists of six layers:



**Fig. 2.** The general architecture of a distributed online social network

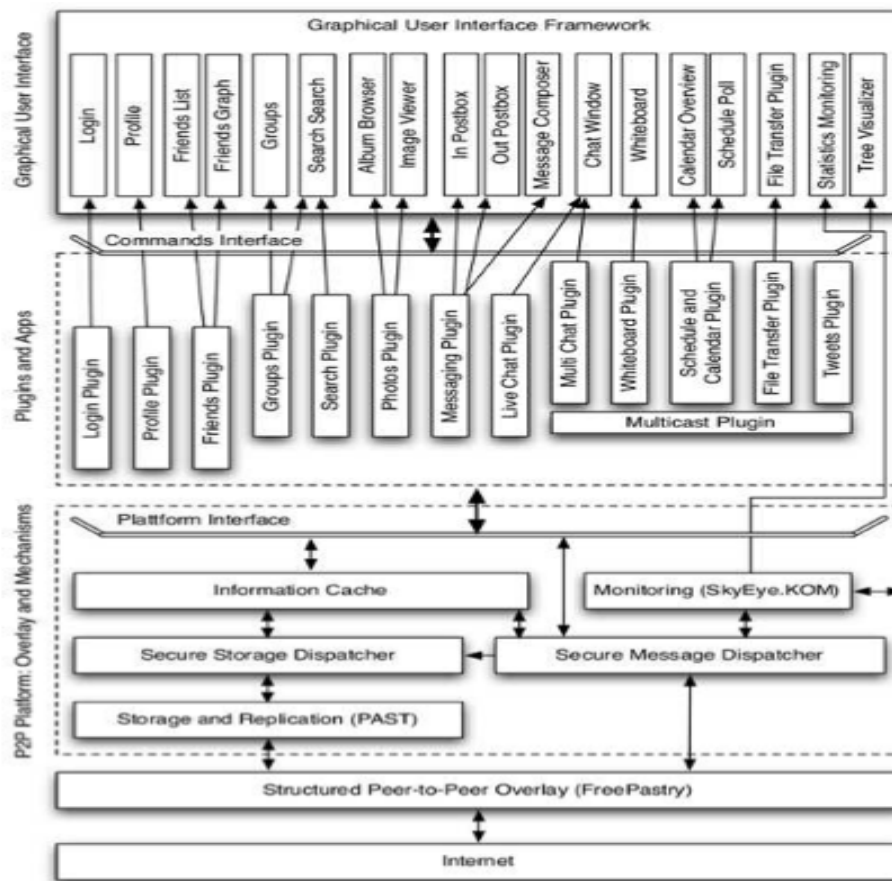


- The physical communication network: this layer provides higher layers the capabilities of looking up resources, routing messages, and retrieving information reliably and effectively among nodes in the overlay.
- Decentralized data management layer: implements functionalities of a distributed or peer-to-peer information system to query, insert, and update various persistent objects to the systems.
- The social networking layer implements all basic functionalities and features as centralized social networking services with the most important ones are the capability to search the system (Distributed search), the management of users and shared space (User account and share space management), the coordination and management of social applications developed by third parties (Application management). In addition, this implements an application-programming interface (API) to support the development of new applications by freelance developers and other third parties as existing API standards – OpenSocial.

#### 4.1 Proposed Decentralized Online Social Networks approaches

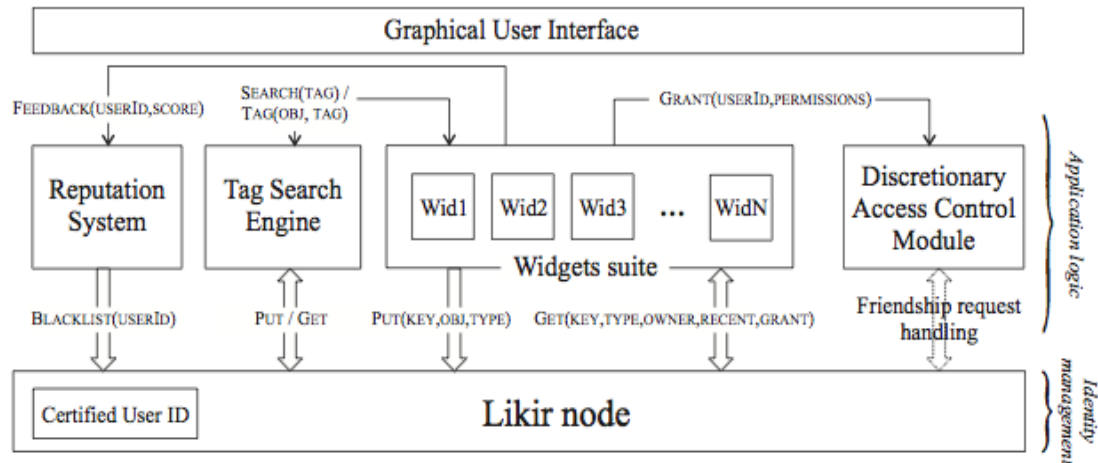
- Safebook: main objective of protecting its users' privacy, integrity, and availability. Moreover, Safebook consists of three major different components, the TIS, matryoshkas, and a peer-to-peer location substrate.
- FOAF: The framework enables users to export their FOAF profiles, store them on dedicated trusted servers. Users query and manage these profiles through open Web-based protocols such as WebDAV or SPARQL/Update
- LifeSocial: primarily aims at keeping social networking services scalable by distributing the load to their users' resources. It is designed with the main premise to leverage on existing and proven components and to create a modular plugin-architecture to assure extensibility.





**Fig. 4. LifeSocial Plugin Architecture**

- PeerSoN: The main properties of PeerSoN are encryption, decentralization, and direct data exchange, which aims at keeping the features of OSNs but overcoming two limitations: privacy issues and the requirement of Internet connectivity for all transactions. To address the privacy problem, it uses encryption and access control coupled with a peer-to-peer approach to replace the centralized authority of classical OSNs
- Likir: aimed to protect the overlay against attacks common to these systems, by embedding a strong identity notion at overlay level. And the main motivation of Likir is to avoid a central data repository, for both the reason to avoid a single point of failure and aggregation of user data.



**Fig. 5. Likir Architecture**

## 5. Specialize Application Centric Decentralized Online Social Networks

### 5.1 Social-based P2P File Sharing

The most well developed initiative is Tribler, which is basically a P2P content sharing system. The system, implemented as social-based extension of the BitTorrent engine and during the registration phase, each participating user is given by email a secure, unique, permanent identifier (PermID). PermIDs are obtained via a public key generation scheme with challenge-response mechanism to prevent spoofing. Existing contacts of a user (a peer with a PermID) can also be imported from other social networks such as MSN and GMail. A peer in Tribler uses many types of caches to store locally any contextual information relevant to its interests and tasted, which they may store various information related to its friend lists, the altruism levels, the preferences of its friends, and meta-data of the files and contents posted in the network.

### 5.2 Shared bookmarks and collaborative search

Diki is a social bookmarking service that allows users to encrypt and share their bookmarks with trusted friends via the Extensible Messaging and Presence Protocol (XMPP). Three design principles for user's privacy: enable data exchange only between trusted friends, not storing any data centrally, and any stored data is encrypted – which use the PGP private key encryption scheme

### 5.3 Micro-blogging

FETHR is a lightweight protocol enabling users to use any existing micro-blogging service to communicate with other users on top of HTTP with near real-time guarantee. According to the FETHR protocol, the publishers (the micro-blogger) push the entire (assumedly small) contents of the messages (tweaks) to the subscribers

## 6. Social distributed systems:

Social distributed systems (SocDS), which provide equivalent functionalities to non-social (and sometimes, also centralized) systems. Distributed/peer-to-peer infrastructure for OSNs provide desirable properties and addressing the shortcomings and constraints of traditional OSNs which use centralized resources provided by OSN service providers

### 6.1 Social DHT: Social Circle:

Distributed Hash Tables (DHTs) provide essential indexing and resource discovering in distributed information systems. Virtual ring routing (VRR) is a DHT style overlay layer approach used to define the underlying network's routing mechanism. It is implemented directly on top of the link layer and provides both traditional point-to-point network routing and DHT routing to the node responsible for a hashed key, without either flooding the network or using location dependent addresses

### 6.2 Storage / Back-up:

A p2p storage (or back-up) system uses the storage space of its participants to increase the availability or the survivability of the data. Also, a distributed storage system is a building block for a DOSN. Many distributed storage systems have the option of sharing stored objects among a group of users. OceanStore – which all the objects are encrypted, read sharing (i.e., many agents accessing an object) is accomplished through sharing of the encryption key.

## 7. Delay-Tolerant Decentralized Online Social Networks:

Delay- tolerant social networks would allow users to benefit from locality. They could find others who live nearby and have similar interests, find or start events in the neighborhood, organize or collaborate for creative or political collective action, found local marketplaces of ideas, goods, or services, edit local information repositories or wikis, to name just a few possibilities.

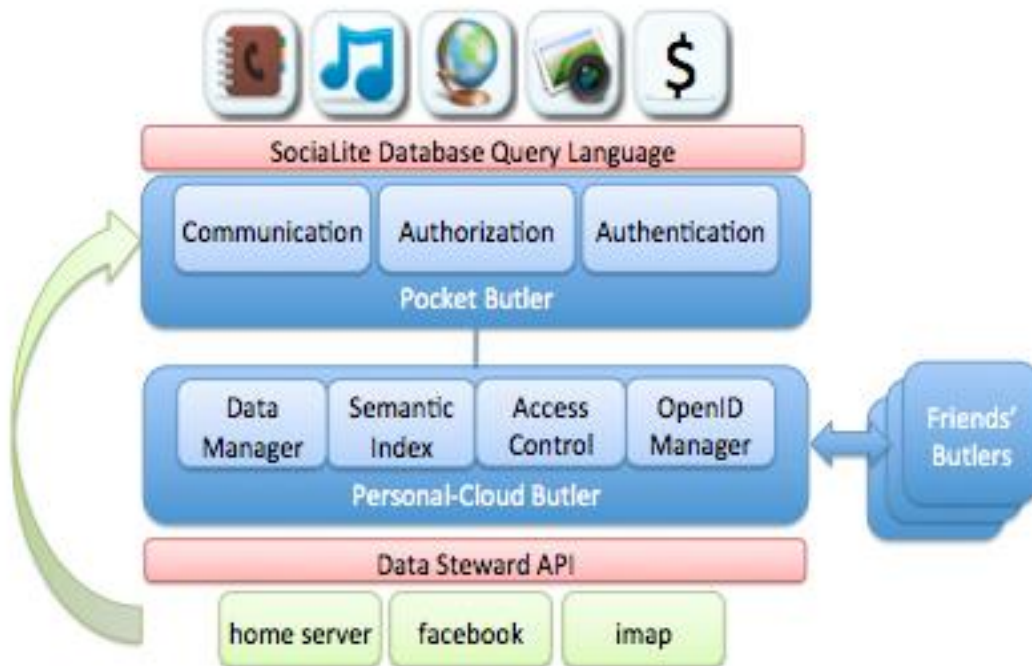
## III. Architectures for Decentralized Online Social Networks

Now we consider the infrastructure and architecture for Decentralized Social Network with PrPI infrastructure and SuperNova architecture

### 1. PrPI:

#### 1.1) Introduction

This part presents PrPI, a decentralized infrastructure that let's users participate in online social networking without loss of data ownership. PrPI, short for private-public, has a person-centric architecture—each individual uses a Personal-Cloud Butler service that provides a safe haven for one's personal digital assets and supports sharing with fine-grain access control.



**Figure 1: The PrPI data subsystem**

#### 1.1.1 Decentralized, Open, and Trustworthy Social Networking

A decentralized, open and trustworthy social networking infrastructure enables people worried about privacy to participate in social networking without reservations.

- **Decentralized:** across different administrative domains. This allows users who keep data in different administrative domains to interact with each other.
- **Open** API for distributed applications allows a social application to run across different administrative domains. Its goal is to allow the same application to be run on different websites separately; that is, each application just operates on data wholly owned by a web portal.
- **Trustworthy** interactions with real friends. The goal is to create a safe haven for individuals to keep all of their data without reservation and to share selected items with different friends. This safe haven will enable new applications since all the personal data are available in one place, and is more convenient for users because they do not have to upload them to different web sites.

#### 1.1.2 Contributions.

This architecture is PrPI, short for Private-Public, as a prototype of a decentralized, open and trustworthy social networking system.

**Personal-Cloud Butlers:** is a personal service that we can trust to keep our personal information; it organizes our data and shares them with our friends based on our private preferences on access control. Butler architecture contains:

- **Semantic index of personal data:** provides a unified index of the data to facilitate browsing and searching of all personal information.
- **Federated storage system.** To take advantage of freely available data storage on the web, the Butler lets the user store their data, possibly encrypted, with different storage vendors if they wish.
- **Decentralized ID management.** The system allows users to use their established personas by supporting OpenID, a decentralized ID management system.
- **Common client proxy:** To support single sign on for applications running on the client device, a Pocket Butler handles the underlying authentication and communications with the Personal-Cloud Butlers.
- **Butler services.** The Butler provides a web-based service that allows friends to log in with their OpenID to enjoy data and services they are entitled to.

**SocialLite Language:** is a database query language that allows easy access to the large amount of data stored in a distributed network of Butler services. SocialLite is an extension of Datalog, a declarative deductive database language. Supporting composition and recursion, this language is expressive enough that many social applications can easily be written by adding a GUI to the result of a SocialLite query. This language hides the complexities in distributing a query to the friends' Butlers.

## 1.2 Federated id management.

The PrPI system utilizes federated, decentralized identity management that enables secure logins, single sign-on, and communication among applications in an environment where Butlers belong to different administrative domains.

Requirements for identity management include authenticating users to Butlers, registering Butlers with the Directory Service, third-party service authentication, and authentication between Butlers and applications.

To this end, OpenID is chosen due to its position as an open standard, extensive library support, availability of accounts, and the ability to extend the protocol easily for PrPI's needs.

### 1.2.1 The Butler Directory Service.

In the future, a Butler service will be associated with each OpenID. The Butler service is to be registered with the OpenID provider, which also serves as its Certificate Authority (CA). To register a Butler with its Directory Service, the owner authenticates himself to the Directory Service using OpenID.

### 1.2.2 User Authentication at the Butler.

Given an OpenID, anybody can look up the associated Butler service and view the information made available to the public. A guest needs to sign on to each Butler service, possibly running in different administrative domains, before using it.

### 1.2.3 Authentication between Butlers.

In the decentralized PrPI architecture, a query presented to a Butler may require the Butler to contact friends' Butlers on behalf of the user. To support single sign-on, we use a PrPI Session Ticket which is a tuple <issuer ID, re-requester ID, session ID, expiration time, issuer's signature>.

## 1.3 The PrPI semantic index.

The Butler keeps an index of personal data and relations which is built with the cooperation of Data Stewards. It enforces access control and presents a programmable inter-face to applications. It also includes a personal homepage with the personalized services and management console so the user can administer and access his data over the web.

### 1.3.1 Semantic Index API.

The PrPI semantic index contains all the personal information (e.g. contacts and locations) as well as the meta-data associated with large data types, such as photos and documents.

The meta-data includes enough information to answer typical queries about the data, and location of the body of the larger data types, known as blobs.

Blobs may be distributed across remote data stores and possibly encrypted.

A unit of data in the system is known as a resource. A resource conceptually is a collection of RDF (Resource Description Framework).

These resources contain much the same information that one would find maintained by a traditional file system such as name, creation time and modification time in addition to keyword and type.

Blob resources contain type and size information about the blob and a pointer to the Data Steward that physically hosts the file.

### 1.3.2 Data Stewards on Storage Devices.

Each federated store that hosts blob data runs a Data Steward, operating on behalf of the user. It provides a ticket-based interface to PrPI applications and hides the specifics about how the blob is actually stored.

### 1.3.3 Butler Home and Management Console.

A Butler's owner can add services to his Butler. These services are accessible via the Butler's web-based homepage. This homepage also includes a management console with which the owner can administer and access his personal cloud information.



## 1.4 Socialite: A Language for a Social Multi-Database.



**Figure 2: The PrPI Butler homepage and photo browser application**

Socialite is an expressive query language for social multi databases. It is based on Datalog which is a query and rule language for deductive databases that syntactically is a subset of Prolog.

Datalog is chosen as the basis of our language for accessing the PrPI social multi-databases. The reasons for choosing Datalog are:

- Datalog supports composition and recursion, both of which are useful for building up more complex queries. Being a high-level programming language with clean semantics, Datalog programs are easier to write and understand.
- Datalog avoids over-specification typical of imperative programming languages. As a result, the intent of the query is more apparent and easily exploited for optimizations and approximations.

### 1.4.1 RDF-Based Database.

The database in the Butler is an unstructured semantic index, meaning that relation schemas need not be predefined. This allows adding new relationships easily. Socialite provides syntactic sugar for RDF by allowing RDF triples to be included as predicates in the body of a rule.

### 1.4.2 Function Extension.

In Socialite queries, user-defined functions may be used to do additional computation on retrieved data. Two types of functions are supported - tuple-wise functions and relation-wise functions.

- Tuple-wise functions can be applied to each tuple of a relation, one at a time.
- Relation-wise functions operate on an entire set of tuples. A relation-wise function is syntactically located on the left-hand



side of a query statement. The function may return either a set of tuples or a single scalar value.

#### 1.4.3 Remote Queries.

Socialite has an According-To operator “[ ]” to allow succinct expression of remote queries. For example, the predicate “P[x](y)”, read P(y) according to x, evaluates predicate P(y) on x’s database.

#### 1.5 Prototype Implementation.

The major infrastructure components include:

- Personal Cloud Butler.
- Data Steward.
- Pocket Butler with Client API for building applications.
- A Global Butler Directory service is created in the meantime to support experimentation (Because we cannot change the OpenID Providers to provide the Butler directory service).

To integrate with OpenID and implement PKI for authentication, SSL, and tickets, we use:

- OpenID4Java library and built-in security and cryptography libraries from Sun.
- Java’s key tool and OpenSSL for generating, signing, and managing certificates and keys.
- Jetty as a web server and Apache XML-RPC for RPC.
- Services communicate with each other over HTTPS and make requests via RPC or REST APIs.
- Custom protocols for efficiently fetching blobs or streaming query results.
- The Butler Management Console is written using JSP and Struts.
- The PrPl index is implemented using HP’s Jena semantic web framework and a JDBM B+Tree persistence engine.

#### 1.6 Application Experience.

The PrPl system enables us to make our personal information, such as contacts, GPS locations and photos, available over the web and on our smart phones. There is a unified contact list, which can be used for sharing all kinds of data.

##### 1.6.1 Sharing Personal Information.

We now describe our experience in building Peops, an Android interface to the Butler service. Essentially, Peops enables users to submit Socialite queries to their Butler service via a graphical user interface. Peops queries a user’s Butler for his list of friends, ranks them by order of tie strength and shows them to the user.

##### 1.6.2 Jinzora: Free Your Media

With the goal of trying to attract real users, we have also experimented in creating a mobile social music experience by leveraging a popular open-source music-streaming web application called Jinzora.

By integrating Jinzora with the PrPl infrastructure, users can stream music to themselves and their select friends, and also share playlists. This de-

sign gives users the freedom to host their purchased music anywhere while enjoying the accessibility typical of hosted online music services.

## 2. SuperNova:

### 2.1 Introduction.

A centralized OSN system uses dedicated servers to store user data (which includes user profile, other content such as pictures and video files, as well as communication with other users). In a decentralized setting, ensuring 24/7 data availability is non-trivial, such enforcement is likely infeasible, and if incentive mechanisms such as reciprocity are used, (rational) peers would pair- up with other nodes greedily to maximize their own data availability, such that the price of anarchy in the system is unbounded

The proposed super-peer based architecture is based on the key observation that it should be possible to incorporate some other forms of incentives in the system. A super-peer based architecture benefits in that:

- It allows the nodes acquiring super-peer status to capitalize on the same in various manners, such as monetize by serving advertisements to the serviced peers, or gain recognition and influence a community.
- It allows for an alternative form of incentive for peers to behave well, in order to gain such super-peer status, in order to in turn benefit from the perks of being a super-peer.
- It allows explicit mechanisms to leverage on well provisioned nodes, which some users may want to contribute, either because of an ulterior incentive that they would like a DOSN to be sustainable (similar to volunteer relay nodes in Tor), or being run by corporate entities who would like to benefit from the advantages of social networking, while not lose control of data.
- It may be run either on end user computers, or on cloud services.
- It helps facilitate new node joins in the system, since new joiners' would typically not immediately know enough friends.

SuperNova architecture allows the system to self-organize by leveraging on heterogeneity, which may arise due to various reasons including altruistic behavior of participants as well as various incentive mechanisms. More specifically, the different user strategies include: choice of nodes where to store its data which can be at friends, at strangers, at super-peers or a combination of the same, as well as the choice of nodes to determine whose data they are willing to store, again including friends and/or strangers; besides nodes' storage capacity and churn, which all together determine the system environment.

## 2.2 Approach.

### 2.2.1 Definitions of Different Entities.

This section describes various entities which are parts of the SuperNova architecture.

- **Profile:** Each node (or user) in DOSN is represented by a profile which can be considered as a medium of expression for the user, as well as user data - which may be public, or protected for limited access to a subset of friends, or private and inaccessible to anyone.
- **Friendlist:** Each user's profile is linked with his friend's profile, which user adds either based on acquaintances or on interest similarity. The collection of all the linked profile is termed as friend list.
- **Storekeepers:** Storekeepers are list of users who have agreed to keep a replication of another user's data so that when a particular node n is down, then n's friends can contact storekeepers to access n's data. Every storekeeper maintains a list of nodes for which they are doing a storekeeping for data synchronization. For a particular node n, every friend in his friend list knows the list of all the storekeepers for the profile (users).
- **Super-peers:** Super-peers are one of the most important entities in the system. Any node may become a super-peer by providing his services to the system in general, and most importantly to new nodes. Super-peers provide storage to the new nodes that don't have enough friends in the network for some initial period. They also maintain and manage different types of services (for example maintenance of user-list) for DOSN by cooperating among themselves.
- **Communities:** Groups or communities are formed based on interest. Any user can create a community. The creator of the community will act as a moderator for the community. Creator (or owner) of the community is the main responsible person for storing all the data and for settling any dispute if it arises in the community due to any posting or because of any other reason.
- **Synchronization:** Logical clock is used for keeping a synch between a node n and his friends and storekeepers. On the same line it is used for keeping a synch between all the community members.
- **Services/Information:** There are two kinds of services, namely - public and protected, available in the network. Anyone can avail a public service. In contrast, if only group members are allowed to access the information related with a particular group or community, then it will be termed as a protected service.
- **List of Directories:**

- **User List:** Every user's id and his public information pointers (the nodes which are storing the public data). This list is maintained by super-peers.
- **Super-peer List:** List of super-peers and their respective services and agreement details. This list is maintained by super-peers themselves.
- **Group (or community) List:** List of communities and pointers for community owner and public content of the community.

### 2.2.2 Bootstrap of new node.

Different phases of a new node  $n_{nn}$  from the point it enters the network till the point it settles down in the network.

- **Initial Phase:** When a new user joins the DOSN, it creates a profile. This initial period is 'Initial Phase' (IP). Time period of this phase could be from few hours to days. When a new node joins the network, he can view a list of super-peers (which is public information) and the list of services they are providing. A super-peer advertises following list of services for the new nodes.
  - **Storage:** How much data they can store.
  - **Agreement time:** For how long they can store.
  - **Time availability:** What times of the day, they can cover for a node.
  - **Content type:** Ready to store what type of content.
  - **Advertisement:** What kind of advertisements will be shown on user's profile by the super-peer
- **Take Care Phase:** Once an agreement is reached between  $n_{nn}$  and a super-peer,  $n_{nn}$  enters Take Care Phase (TCP), in which the super-peer helps improve  $n_{nn}$ 's data availability and thus acts as a storekeeper. The take care period is the time when the new node tries to find and establish friendships in the system. Super-peers keep track of new node's geographic location and up-time statistics. Each super-peer maintains two kinds of pools to facilitate users in finding other nodes:
  - **Time Track pool (TT):** As part of agreement, during take care phase, the super-peer tracks new node's up and down time.
  - **Stranger pool:** Every super-peer maintains a stranger pool along with TT pool.
- **Settled Phase:** A node may extend its take care

phase with a super-peer if he is not able to get enough storekeepers of its own to get desired data visibility. Once a node has enough storekeeper(s) (other than super-peer) to manage his data, it can relinquish super-peer's services.

### 2.2.3 Data Synchronization and Updates.

This section describe merging, copying and updates of data between storekeeper(s) which is one of the important aspects of DOSN.

- **Data Storage:** When a node  $n$  goes down it pushes all its latest data to all the up storekeepers. When node  $n$  comes up, it requests all the storekeepers (whoever is up) for the latest data. It accepts the data from the storekeeper, which is having latest data.
- **Updates:** When a friend  $f$ , posts a comment on node  $n$ 's status, an update request is send to node  $n$  (if it's up) with a timestamp (logical timestamp). If  $n$  is down, requests are made to all the storekeepers who are up, about the updates. Every update request comes with the timestamp. A node  $n$  (or storekeeper(s)) might receive multiple updates at same item, in such cases, node  $n$  (or storekeeper(s)) uses timestamp to resolve any conflicts.

### 2.2.4 System Model.

Let  $G(N, E)$  be the DOSN graph, where  $N$  represents nodes (or users) of the network and  $E$  represents the set of edges between any two nodes  $n_i$  and  $n_j$ . Let represent the global stranger pool managed by all super-peers and  $SP$  the list of super-peers. To increase the availability, a new node can share his data with a particular super-peer  $n_{sp}$  s.t  $n_{sp} \in SP$ . A node can get good availability with the help of friends if many of them are ready to act as storekeepers. We define friends of node  $n_i$ . A node might be able to convince some strangers for storekeeping to boost his availability. Let  $\gamma_i$  represent the set of storekeepers for node  $n_i$ , then formally it can be represented as

$$\sigma_i = n_i + \sum_{sk=1}^{SK} n_{sk}$$

$$s.t \forall n_{sk} \ n_{sk} \in \gamma_i \text{ or } n_{sk} \in \psi \text{ or } n_{sk} \in SP$$

Let  $T_i$  represents the set of time intervals for which node  $n_i$  is online. In decentralized system if the data is not stored at other than the owner of the data, then data availability is equal to the time units for which node  $n_i$  is up, that is  $Avl(n_i) = |T_i|$ . If a node is able to store data on other nodes as well, then availability of node  $n_i$  can be defined as

$$Avl(n_i) = |\bigcup_{al=1}^{AN} T_{al}| \text{ s.t. } \forall n_{al} \in \sigma_i$$

### 2.3 Evaluation.

In this section we evaluate using various metrics how a DOSN based on SuperNova architecture will perform for different mix of user behaviors.

#### 2.3.1 Network.

We use the social network graph from a subset of the DBLP co-authorship graph as a representative social network. Specifically, we use the giant connected component of co-authorship graph from DBLP record of papers published in conferences between 2004 and 2008, comprising of 273798 unique authors. Each author represents a node in the network. The graph has an average degree of 6 with diameter of 23.

#### 2.3.2 Trace Generation.

Considered following three different factors for the trace generation:

- Location: We consider twelve different (time zone) locations, with each location differing by 2 hours from its adjacent locations.
- Weekday Behavior: We consider six types of behaviors during the weekday.
- Weekend Behavior: We consider four types of behaviors for the weekend.

#### 2.3.3 Various parameters for Storekeepers.

Different users have different behavior and to capture individual behavior is difficult so we generalize it for the whole network.

- $P_{fd}$ : This parameter decides with what probability a friend/neighbor is ready to accept the data of a node.
- $P_{sd}$ : The percentage of nodes which wants to participate in stranger pool.
- $P_{as}$ : With what percentage a stranger is ready to accept another stranger's request for storing the data.

The table summarizes the various combinations that we analyzed by selecting different percentage values for each parameter.

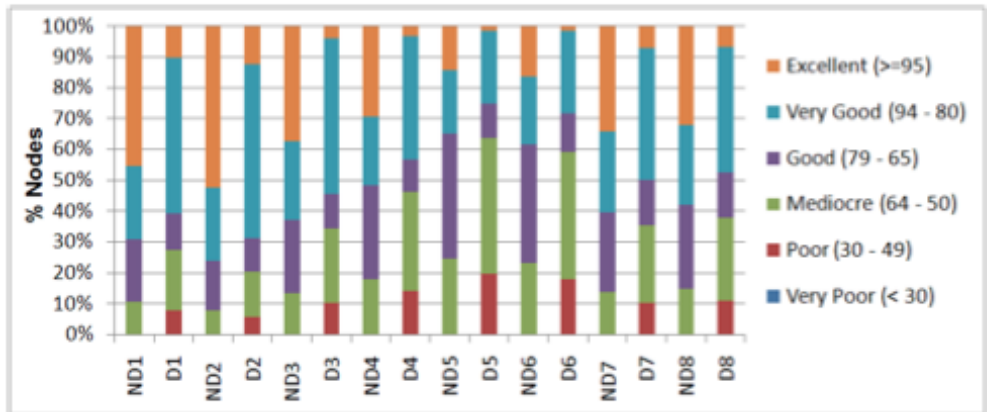
#### 2.3.4 Various node strategies for storage.

A node in general can store its data using any of the four following schemes depending on various factors like if its social neighbors are already overloaded and whether he is (not) willing to store data on nodes run by strangers.

- $S_{nns}$ : A node is able to store his data on some of his neighbors as well on strangers.
- $S_{nn}$ : A node is able to store data only at his neighbors. A node might not be interested in storing data at strangers because either he is not trusting or he is not able to convince strangers.
- $S_{nsp}$ : A new nodes store his data on one of the super-peers after agreement.
- $S_{ns}$ : A node might not be able to get help from any of its neighbors for storage. However a node might go for a scheme where he is able to convince some strangers for storing the data.

We define availability time as the ratio of the number of time units for which a node n's data is available to that of total time unit of the simulation. We categories the availability (Avl) into 6 classes namely (names are intuitive for performance)

- Excellent ( $Avl \geq 95$ )
- Very Good ( $95 > Avl \geq 80$ )
- Good ( $80 > Avl \geq 65$ )
- Mediocre ( $65 > Avl \geq 50$ )
- Poor ( $50 > Avl \geq 30$ )
- Very Bad ( $Avl < 30$ ).



**Fig. 1.** Performance of different parameter combinations for Deviation (D) and Non Deviation (ND)



We measure the percentage of nodes falling under each availability category with three different scenarios.

- To highlight the effect of super-peers we compare our super-peer based architecture with a flat scheme.
- A comparison between an ideal (or best) case and worst is also presented to reflect the effect of cooperation and selfishness respectively.
- Nodes might be interested in getting availability in terms of the time when their friends are online vs. 24/7 availability.

In evaluation of all the three cases - along with measuring cumulative availability, we try to gather information about what percentage of nodes are getting what percentage of availability.

## IV. Conclusion

Decentralized social networks have the potential to provide a better environment within which users can have more control over their privacy, and the ownership and dissemination of their information, which can solve existing problems of centralized system. More importantly, a decentralized approach to online social networking breaks the boundaries between social networking sites by providing users more freedom to interact with each other. However, one of major challenges of realizing decentralized online social networking is its adoption by users who are already participating in existing social networking sites will need to migrate their data to decentralized social networks to break away from the traditional data silos offered by the current social networking sites. Moreover, a newly developed decentralized social system is less appealing to new users due to its lack of benefits, while the system itself relies on a certain critical mass of participants before it can offer its users any significant values. Also, various performance issues, mostly related to the availability, latency, and throughput in data access due to data encryption and replication, of these decentralized social applications have still yet to be investigated carefully to compare with their existing centralized approaches. Despite the above challenges, we believe that development and research on DOSNs still are very important and have significant impact. In this report, we mentioned two types of decentralized infrastructure for social networks as PrPI and SuperNova that provides flexibility to individual users to choose their strategy in terms of where to store their data, as well as whose data to store. While we do not investigate the role of trust and reputation explicitly in this report, recent trust model which correlate node's high reputation with them being more influential custodians of the community's tasks provides a premise to carry out. Hence, to attract users use decentralized online social networks, developers should build a good user interfaces including complete functionality of OSNs as tools for importing and exporting data, and ease of setting up the software, allowing for direct data exchange between devices (as music, photo, documents, etc. ), enabling ad-hoc social communities, data locality, and delay-tolerant social net-works.

## V. Reference

1. Decentralized Online Social Networks: Anwitaman Datta, Sonja Buchegger, Le-Hung Vu, Thorsten Strufe, Krzysztof Rzadca
2. Decentralization: The Future of Online Social Networking: Ching-man Au Yeung, Ilaria Llicardi , Kanghao Lu, Oshani Seneviratne, Tim Berners-Lee
3. PrPI: A Decentralized Social Networking Infrastructure: Seok-Won Seong, etc.
4. SuperNova: Super-peers Based Architecture for Decentralized Online Social Networks: Rajesh Sharma etc. (google, existing architecture)