Open Social Net

Andrew Hitchcock

Jimmy Lewis

Operational Concepts

This product is intended to provide a decentralized social networking system that allows users to have data mobility and the control of the proliferation of their data. The project would be an open software allowing hosts to create new networks for users (based around interests, jobs, schools, or communities). There would be tools available for hosts to limit access to their network based on various criteria (such as requiring them to have a certain e-mail address). The users could join any networks they choose, with different visibility settings for each network. They would have a single profile kept on a "home network" and, based on privacy settings, select parts of their profiles would be replicated to other networks in which they are members. Objectives for this software would be to create a social networking site built on a decentralized model in order to allow for flexible roaming profiles and random network outages without loss of data. Additionally, users would be able to maintain most of their profile and settings, even if one network is shut down. Users would also be able to use this product to aggregate information from other networking sites into one central location.

System Requirements

Server operators would need a web server supporting the product development language and database system (TBD), and any requirements needed for individual modules.

Users would need a web browser, likely with support for asynchronous JavaScript requests (AJAX) and cookies.

System and Software Architecture

The basic core of the application will provide basic services relating to user logins, profiles, network settings, as well as providing background data replication and enforcing data consistency. This functionality would be expanded by a pluggable module hierarchy (see Fig. 1), providing functionality to add services such as exchanging data with other sites or sources.

User profiles will be stored in their entirety on the server corresponding to their "home network." This data would be replicated as needed to other network servers (Fig. 2), allowing for data distribution to occur in a manner such that traffic is localized to the network from which a request originates.

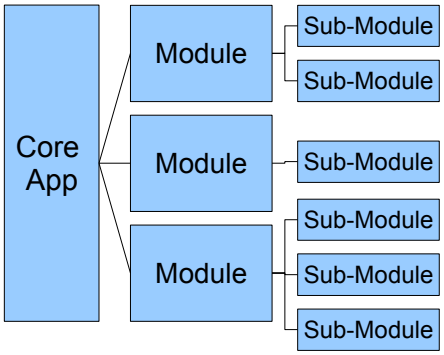

Fig. 1

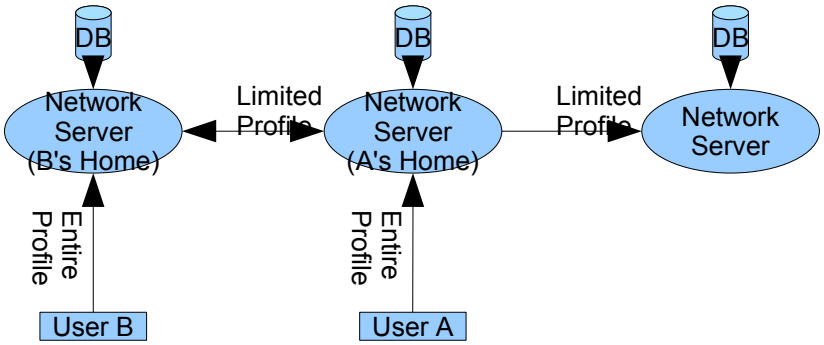

Fig. 2

For example, in Figure 2 let Users A and B both belong to each other's networks but have separate home networks. If User A makes a request to his home network for User B's profile, he might only receive a limited profile because B's profile is limited on that network; however, if User A makes the request to User B's home network (of which User A must be a member) he will receive the entire profile. This also allows for users to have distinct privileges between their network profiles – based on network membership.

Modules are intended to add existing functionality from other sites, such as importing and exporting data between other sites. Services might include other networking sites, photo sharing, etc...

We've considered using PHP for the core with MySQL as a database engine, due to widespread availability of both, but this decision has not been finalized. XML will likely be used for inter-server communications. Other technology considerations include ASP.NET, Ruby on Rails, MSSQL, and PostgreSQL.

Lifecycle Plan

We intend on using a hybrid lifecycle model based on design-by-schedule and sub-projects. This is due partly to the time limitation, and the modular design allows us to easily change the feature set depending on actual rate of development.

We think that two weeks will be required for high-level planning and to get started on detailed specifications. Another week should be spent determining our own API for module development. At that point we can split into sub-projects, beginning with the core backend and frontend sub-projects. As those are completed, members can begin module development based on a prioritized list to be determined by the development team.

This project will likely require 6-8 developers, preferably with experience in the technologies used, specifically a server-side scripting language, database skills, and web design. As the technologies are not yet determined, the decision may depend on developer experience.

Feasibility Rationale

We believe that this project can be completed in the alloted time due to the modular design. Since the core application is not required to support a large functionality, the optional modules can be developed on an as-feasible basis. Some of the risks include an inability to solve the following problems: replication design while ensuring consistency; permissions issues between networks; consistent data proliferation according to privacy settings; inter-network security; and limitations imposed by foreign services (e.g. for module development).