Enabling Intra-Community and Inter-Community Support in Lean Societies

Laurie Butgereit
Port Eliabeth

Abstract
Communities in developing societies are often in need of support in terms of obtaining knowledge and information. Sometimes this support comes from outside the community (inter-community support) and sometimes this support comes from within the community (intra-community support). Inter-community support happens when one community of people assists another community of people. Intra-community support happens when a community helps itself. An example of inter-community support would be a community of health care workers which assists a community in a specific physical location. An example of intra-community support would be a community of teachers who assist each other in understanding new curricula. The open source project Communities on Fire enables both inter-community and intra-community support via synchronous collaboration using a brandable and configurable Android application. This project was instantiated for both an inter-community support project (in the case of mathematics tutoring) and an intra-community support project (in the case of a mutual support study group for PhD students). The results of these two instantiations are present in this paper and will be of interest to researchers wanting to provide knowledge and information to communities.

Keywords
Inter-community, intra-community, support
Introduction

Communities in developing societies are often in need of support in terms of obtaining knowledge and information.

In this respect, the term community does not just refer to a group of people living in a certain physical area such as a certain village. It can also refer to a group of people which share some common characteristic or interest such as a community of teachers or a community of knitters the location of which spans certain physical areas. The term has also often modified with the adjective virtual as in a virtual community to describe a group of people who only communicate using Internet mediated communication methods (Howard Rheingold 2000, Song 2009).

The term developing societies and developing world are out of date. In February, 2014, Dayo Olopade, op-ed contributor to the New York Times, argues that it is time to find a replacement for the term developing world (Olopade 2014). In an article, Olopade quotes Bill Gates in saying that “...the terms developing countries and developed countries have outlived their usefulness”. It can be argued that all countries need development and even countries which are traditionally classified as first-world countries (such as the United States) can benefit from continued development. It can be argued that the current roll-out (whether or successful or not) in the United States of affordable health care (also known as ObamaCare) can be seen as a type of continued development.

Olopade suggests that the terms lean society and fat society be used in their places. Although these terms often correspond to obesity rates in those societies, she explains that a lean society approaches problems with scarcity in mind while a fat society approaches problems with abundance in mind. An example of these different approaches would be a problem of lack of banking facilities in an area. In a fat society, a solution would be to build a bank branch and install ATMs with high speed connectivity to central banks. In a lean society, such as Kenya, on the other hand, the solution was the creation of the successful M-pesa mobile banking facility. In an interview on Internet radio station WNYC, Olopade explains that lean societies emphasize the positive efficiency and creativity which is generated by resource scarcity (Lopate, Olopade 2014).

In many situations, one of the scarce resources is information or knowledge. Lack of information, lack of knowledge, lack of education, etc, is a common problem in lean societies.

Very generally, there are two solutions to this scarcity of information problem. The information, knowledge, or education can come from outside the society or community or it can come from inside the society or community.

In the expressions inter-community support and intra-community support the prefixes inter- and intra- (with or without the hyphen) sound very similar and are often confused. The prefix inter- means between and can be seen in expressions such as international trade (trade between nations) and intercontinental air travel (air travel between continents). The prefix intra- means within and can be seen in expressions such as intramural sports (sports competitions within an organisation) and intranet (computer network within an
organisation). Examples of inter-community support include projects such as the Dr Math project where a community of university tutors assists a community of primary and second school pupils (Botha, Butgereit 2012) and Doctors Without Borders where a community of medical professionals assists distressed communities around the world (Doctors Without Borders 2010). Examples of intra-community support include Alcoholics Anonymous where participants attempt to solve their common drinking problems by assisting each other (Alcoholics Anonymous 2014) and typical study groups at universities where students form their own groups to support each other in their studies.

Because of these two distinct types of assistance (inter-community and intra-community), it is often necessary for software deployed in such situations to have different user interfaces or multiple user interfaces to cater for both situations.

This paper describes the Communities on Fire project which provides both intra-community support and inter-community support to assist in lean societies.

**Synchronous vs Asynchronous**

The terms synchronous and asynchronous are often consider to by antonyms but both terms can be used to describe the same event if viewed from different perspectives. In general, the term synchronous communication refers to communication where time is important and the term asynchronous communication refers to communication where time is not important. But these definitions need to be clarified with the expression important to whom or to what.

When computer scientists or information technologists refer to synchronous communication or asynchronous communication, they are usually describing the communication in terms of the computers or digital devices involved. In such a situation, a computer scientist would say that the HTTP (Hyper-Text Transfer Protocol) is a synchronous protocol. When a digital device such as a tablet makes an HTTP call to a server, the server must respond back within a specific amount of time. It is, therefore, a synchronous protocol. When a digital device however, sends an XMPP (eXtensible Messaging and Presence Protocol) message to a server, the server is not required to respond with any message. It is, therefore, an asynchronous protocol.

Social scientists and linguists, however, will describe communication in terms of the human beings involved and not in terms of the digital devices. When a human being receives an XMPP message (this is also called a chat message), the human being is socially obligated to reply within a very short period of time. In the eyes of a linguist or social scientist, XMPP is a synchronous protocol (Durham 2009, Tagliamonte, Denis 2008). In addition, if a person posts a question on a bulletin board system using HTTP, the person is not expecting an immediate reply. In the eyes of a linguist or social scientist, a bulletin board system running under HTTP is an asynchronous protocol.

These two terms will be used in paper and their definitions will hopefully be clear from context.

---

ISBN: 978-0-620-63498-4
Pages 141-150
Communities on Fire

Communities on Fire is an open source project which enables both inter-community support and intra-community support. It supports synchronous (in the eyes of the linguist or social scientist) communication among a group of cooperating people (intra-community) and in situations where one community of people is assisting another community of people (inter-community).

At the center of the Communities on Fire project is OpenFire. OpenFire is an open source real-time collaboration server which uses XMPP (eXtensible Messaging and Presence Protocol) protocol for instant messaging (Jive 2013). OpenFire enables one-to-one, one-to-many, and many-to-many synchronous communication among a community of people.

The one-to-many format messaging is suitable for inter-community communication where a community of some type of expert such as medical professionals or mathematics tutors provides assistance to another community of people. The many-to-many format messaging is suitable for intra-community communication where everybody in the community can communicate with everybody else in the community. The one-to-one format messaging can be used in both cases where privacy between parties is required.

The OpenFire project provides an XMPP server, client software for Linux, Windows, and Mac, web components, and a Java library (Jive 2013). At the time of writing, however, OpenFire does not supply a complete Android application.

Having said that, there are Android XMPP (also known as Jabber) applications which can successfully communicate with the OpenFire server. These include Xabber, SecuXabber, Trillian, and Bruno (all easily found on the Google Play Store). While these are useful, they all require a certain Internet-savey on the side of the users. The users must understand the concept of server names and port numbers, creating user accounts on a server and the importance of remembering their usernames and passwords. In addition, these specific Android applications were primarily designed for one-to-one communication and although the one-to-many and many-to-many were supported, there were certain difficulties which needed to be overcome by the user. In addition, these Android applications are not brandable.

The research question for the Communities on Fire project was, therefore:

Can a brandable and configurable Android application be written to communicate with OpenFire in one-to-one, one-to-many, and many-to-many configurations which would support both inter-community and intra-community communication for users who are Internet-naive?

The research question can be broken down into the following sub-questions:

1. How can a brandable and configurable Android application be written to communicate with OpenFire?
2. How can the application communicate using one-to-one, one-to-many, and many-to-many formats thereby enabling both inter-community and intra-community communication?

3. How can the Android application be used by Internet-naïve users?

It is important to note that research question dealing with Internet-naïve users only refers to the Android application.

Research Methodology

Information Systems research can be divided into two paradigms: design science and behavioral science. Behavioral science has its roots in natural science and attempts to provide a picture of reality by creating, developing, and justifying theories. Design science has its roots in engineering and attempts to provide solutions to problems by creating artifacts (Hevner, March et al. 2004).

One of the fundamental requirements of Design Science Research is the creation of an innovative artifact to solve an important problem. The Communities on Fire project attempts to help solve an important problem of lean communities needing information and knowledge. In view of the fact that an innovative artifact would need to be developed in order to solve this problem, Design Science Research would be appropriate.

In order to successfully create and develop such an artifact, the General Design Cycle of Design Science Research (Oates 2006) is defined to have five steps: 1) Awareness 2) Suggestion 3) Development 4) Evaluation 5) Conclusion.

The awareness step identifies the problem to be solved when the researchers become aware of the problem at hand. The suggestion step provides possible solutions to the problem. The development step implements these suggestions. The evaluation step tests and evaluates the implementation. The conclusion step is the final conclusion of the research project.

When the Communities on Fire project was initiated, there was no clear view on how this innovative artifact could be created. An interactive approach was necessary to try different tools and different methods. The General Design Cycle of Design Science was used because of the iterative nature. The General Design Cycle is, indeed, a cycle and the steps can be iterated over numerous times.

Mobile XMPP Libraries: Smack vs ASmack

In the early iterations of the General Design Cycle, the Communities on Fire project was introduced to the OpenFire synchronous communication platform. As described on the OpenFire website (Jive 2013), the OpenFire project includes a message server, chat clients for Windows, Linux, and Mac, and a Java library, Smack, which provides all the XMPP functionality. Unfortunately, there were no Android clients available. The Java library is open source and can be downloaded and compiled. Unfortunately because of various licensing problems, Smack will not compile using the Android development kit. A fork was made in the Smack sources by the open source community and ASmack was created.
At the time that development was started on the Communities on Fire project, it was necessary to include the entire sources to ASmack and recompile ASmack sources with the Communities on Fire sources. However, a release on the OpenFire website dated April, 2014, reports that they have repackaged Smack for Android and have released (a)Smack 4.0.0 (Flow 2014). This release has not been tested or used by the author of this paper.

This paper will now handle each of the three research sub-questions and explain how they were answered.

Configurable/Brandable Android Apps

Android developments insists on a complete separation between the user interface and executable code (Sheusi 2013). The user interface is supplied in XML (eXtensible Markup Language) and the executable code is written in Java. This complete separation of interface and execution made it easy to configure and brand Communities on Fire. Figure 1 shows three different examples of the Communities on Fire user interface. Besides being branded with different colours and icons, the various strings are easily changeable, and, as can be seen by viewing the action bar on top of the application, even the various execution options can be configured in and out. It is also easy to include options for advertising (as can be seen in the far left image of Figure 1) if the project wished to attempt to generate some revenue.

Using the ease of configuration along with the ASmack sources provides a positive answer to the first research sub-question:
A brandable and configurable Android application can be written to communicate with OpenFire server using ASmack by leveraging the fact that Android has an complete separation between the user interface and the executable code.
One-to-One, One-to-Many, Many-to-Many

By default, OpenFire supports one-to-one communication between two users. However, in addition to that, OpenFire supports the concept of conferences and workgroups. A conference is a chat room. Any message which is sent to a conference is then forwarded to all people who are active in the conference. In Figure 1, the conference option can be seen in the third example and is indicated by the multiple profile silhouette in the action bar. This supports intra-community communication where all people who have the application can easily communicate with each other.

A workgroup is a queueing system where a group of participants can be assigned to be members of a workgroup. When messages are sent to a specific workgroup, then they are forwarded to one specific member of the group who is currently logged in depending on various algorithms defined within OpenFire itself. This supports inter-community communication where one group of people is assisting another group of people.

In support of workgroup communication, OpenFire also releases the Spark client software which is available for Linux, Windows, and Mac. In many situations of inter-community communication, the group supplying information has very few members and the group requiring information has many members. Such is the case with the Prof Π (Prof Pi) project where tutors are often chatting with twenty to thirty pupils at the same time. The Spark client is designed for traditional computer workstations and laptops and provides easy swapping between conversations.

In the case of Communities of Fire project, it was decided that the one-to-one and one-to-many (workgroup) facilities were mutually exclusive with the default being the workgroup facility. In Figure 1, the third example had both many-to-many and either one-to-one or one-to-many (indicated by the two different silhouettes in the action bar). The first and second examples only had one of the various facilities configured in with no options in the action bar.

Using the ease of configuration along with the default one-to-one chat facilities of OpenFire, the workgroup facility and the conference facility, the second sub-research question can be answered positively:

The Android application can support one-to-one, one-to-many, and many-to-many communication by using the workgroup and conference facilities of OpenFire thereby enabling both inter-community and intra-community communication.

Naïve Internet Users

The use of OpenFire (and XMPP in general) requires a certain general knowledge about the Internet. This includes the use of fully qualified domain names, port numbers, usernames, and passwords. Often, the members of lean (developing) societies have not had the opportunity to gain this information. This is one of the reasons that XMPP clients available on the Google Play Store (such as Xabber, Trillian, Bruno, etc) were not used in this project.
In order to solve this problem, the fully qualified domain names and port numbers required for the XMPP protocol are included in the brandable/configurable Communities on Fire implementation. In other words, in the three examples in Figure 1, the fully qualified host names and the port numbers were configured into the application at the same time that the icons and colour schemes were configured.

When Communities on Fire starts executing for the first time on an Android phone, it asks the user for his or her name. The application then uses a combination of device specific identifiers (such as the MAC address of the device or the IMEI number) as the password for the account. This precludes the requirement of the user having to remember such details. The application attempts to create an account on the server. If there happens to be a conflict, an appropriate error message is sent to the user asking him or her to slightly modify his or her name by adding a middle name or using a middle initial. The successful username is then stored locally on the Android device so that the user does not need to type it in again.

Obviously, there are other things necessary to assist naïve Internet users and Communities on Fire does not address all the potential problems. However, the third research sub-question can be partially answered:

An attempt has been made to make it easy for naïve Internet users to use the Android application by preloading many of the Internet configuration into the brandable Android applications and generating passwords from the MAC address and IMEI number.

Instantiations

Two instantiations of the Communities on Fire project were used to test the platform. The first instantiation is Prof Π (π). Prof Π is an example of inter-community support. It is a mathematics tutoring project which involved two different communities of people. One community of people are primary and secondary school pupils who download the Prof Π from the Google Play Store and start asking questions about their mathematics homework. The other community of people, however, are undergraduate students at the University of Pretoria who act as Prof Π and assist the pupils with their mathematics homework. The tutors use the Spark interface to OpenFire and the pupils use the Android app which has been branded in bright yellow colours as shown in the far left portion of Figure 1. Because the project dealt with minor children, the many-to-many communication was not implemented.

The second instantiation did not have a specific project name. It was a mutual support project for PhD students attending an Internet based taught module. The students and lecturer spanned three countries. By using a specific implementation of Communities on Fire, the students could easily stay in contact with each other and support each during the course of the taught module.

Testing and Evaluation

At the time of writing this paper, although Communities on Fire has been configured and deployed in a number of projects, it is currently only undergoing live-world testing in one configuration – the Prof Π project.
In configuring these different projects, the same Java code base is used in all cases. Nearly all of the user interface is the same in all projects. Only the logos, colour schemes, connectivity details, and some configuration options change. The configuration of a new project, including the generation of documentation, takes approximately one hour.

Looking at specific log files from the Prof Π project and judging from the type of mathematics questions the pupils are asking of the tutors, the pupils range in age from mid-primary school (asking about multiply numbers and prime factors) to first year university bridging mathematics (asking about trigonometry). In the case of the youngest users, they did not appear to have any problems using the app and it downloaded easily from the Google Play Store.

Conclusion

It can be argued that the terms developing country and developing society are outdated. One can argue that even countries which are traditionally considered to be first-world developed countries can benefit by continued development. Health care reform and banking reform are just two examples of continuing development which can occur in first-world countries.

The terms lean society and fat society (as defined by Olopade) describe whether a society tackles problems from the standpoint of scarcity or from the standpoint of abundance. The two approaches often differ.

Projects which assist communities in lean societies can be classified as either inter-community support projects or intra-community support projects. Inter-community describes a situation where one community of people assists another community of people. Intra-community describes a situation where a community of people assist themselves.

The Communities on Fire project provides facilities for both inter-community support and intra-community of support using the various OpenFire tools (XMPP server, Spark, Smack, and ASmack). An additional component was developed which was brandable, configurable Android application designed to be used by Internet-naïve users. Communities on Fire supports synchronous communication in one-to-one, one-to-many, and many-to-many formats.

Communities on Fire has been configured for a wide variety of projects and is currently being live tested in one such project.

References


