PROTOCOL FOR SYNCHRONOUS CONFERENCING

Last updated 2007

Carlo v. Loesch
PSYC architect
symlynX

Whitepaper on PSYC

Abstract

The Objective

PSYC is a flexible protocol to set up a worldwide distributed messaging infrastructure mostly for text-based conferencing (popularely termed chatting). It was developed with the goal in mind to replace the Internet Relay Chat protocol, which has reached its limits. It however extends many of the concepts commonly found in chat and community systems and is utilizable for purposes such as multimedial communication. Existing conferencing systems can easily be interfaced into the PSYC network since the complexity of the protocol resides on the client side.

The Strategy

Basic principle is to not even start setting up a world wide directory service on the presence of people and existence of conferences, but rather borrow the URL concept from the web: assign locators to people and rooms, then let the client programs set up direct or multicast connections between each other. The servers have a powerful role in coordination and control, yet aren't troubled with directory, traffic or routing problems. Matching this new role distribution for clients and servers PSYC provides a minimalistic conference control system which allows for total profiling of conference politics (Who's to join a group? Who's allowed to listen in? etc.) without affecting routing (with or without multicasting ability).

What *is* a chat system?

Introduction

A real-time synchronous conferencing system,

aka a chat system, is a form of group communication where people sitting at a networked computer in different locations on the planet get together in a virtual room and speak with each other typically by typing text, at least these days. The word "chat" alludes to the typical relaxedness of socialisation going on in chatrooms, but don't be misled by it: chat systems can very well be suited as an additional tool for business communication.

What we want

Decentralization is cool

Decentralized structures are commonplace on the Internet, since they scale best and avoid bureaucracy by design. For asynchronous communication we have e-mail servers and clients, for information gathering there is the WWW. Anonymous FTP is popular for publishing of files and software. But for synchronous conferencing there is no comparable protocol available.

What we have

talk, msend, SMTP SEND ..

talk and similar protocols do operate using the client/server principle, but are aimed at machines where people sometimes log into, instead of orienting themselves directly at the people.

Systems with a central server

Easy, flexible, everything under control, typically proprietary, very popular in the commercial arena.

But:

- How many people can a single server take?
- How are you supposed to meet people again once you met in this or that system. Always login into all systems at once?
- And does it make sense (network topologically) to let two austrians chat on an australian chat server?

Buddy Lists

Buddy lists as provided by AOL and ICQ use proprietary protocols and are based on central database servers. Privacy is also an issue here, as unsolicited promotional messages can be.

Internet Relay Chat

This is where IRC came in back in 1987, and slowly advanced to become the biggest chat system on earth. No matter where you are entering IRC from, you always meet your friends and colleagues since they always use the same nicknames and channels (rooms, conferences). Additionally IRC is capable of telling you when certain nicknames, your friends and colleagues, show up on the net. Unfortunately IRC suffers of inscalability, bureaucracy, hackability.

Use internet telephony for chatting?

SIP is too simple, H.323 has less conference control features than IRC. It's a better idea to launch telephony from a well-conceived messaging system than the other way around.

So what about Jabber™ aka XMPP?

What's good about it

It's good that Jabber has an open distributed network structure and the concepts of identification and location are quite similar to PSYC.

What's different

Jabber is designed as an instant messaging system with a focus on one to one communication. Efficient distribution to groups of people is missing although every update of presence information goes out to a big group of peers! No surprise Jabber already encounters massive scalability issues. Limiting the amount of permissible contacts per person cannot be the answer. Additionally the XML format is enforced, which suggests a certain order and structure. But on the downside it makes framing impossible, that is to be able to know where a packet ends without having to analyze all data as it arrives. Also transfer of files can only be achieved with cumbersome encodings. And then there are a thousand little oddities that fippo had to deal with while he worked his way through the implementation. You can find some more hints like that in the jabber2psyc help page.

Why is it so hard?

What's the problem?

The reason why there isn't a simple and clear solution to the problem yet lies in the complexity of **modeling people** as well as in the **management and routing of conferences**.

How do we get over this?

First of all, throw out the directory service by introducing a global addressability instead. Then make someone really in charge of conference control, and rather ensure that the administrative power over chatrooms is distributed over the whole world according to where the people using them are. We'll get to both of these points now:

Uniform Network Identifications & Locations

psyc://psyc.kanzleramt.de/~gerhard

This is how people modeling works with PSYC: UNIs like **psyc://psyc.kanzleramt.de/~gerhard** induce a client implementation to connect to the psyc.kanzleramt.de server. A program answers on that server that acts on behalf of chancellor Schröder. It will accept the message and maybe deliver it to wherever Gerhard is located, be it in China or Tuvalu. It might aswell be instructed to allow for a direct peer to peer connection between the two clients. In this case it would require a UNL, such as **psyc://bill.pcnet.whitehouse.gov:34209** By the way: ve stands for "virtual environment".

advantages of addressability

UNIs (The term *resource* in URI is just slightly unfriendly to humans) give you a huge amount of benefit: They can be interspersed into the web, appended to e-mail signatures, or appear on business cards, papers and magazines. It will always either put you in touch with the person right away, or let you leave a note on the answering machine.

nicknames *are* owned

The UNI allows for easy authentication schemes which guarantee the identity of a given person, so you won't be able to "fake" another person by taking his nickname, as it is feasible on IRC.

Applications of UNI+UNL messaging

answering machines

Should Gerhard be busy playing golf with Bill his *home-server* will accept messages for him, or simply "call back" as soon as the chancellor is back at the keyboard. Or it could be voice control. Simply to be able to send messages "around the corner" brings a lot of potential applications. Think of BITnet's "tell" command, you could execute it from within all kinds of automations and scripts, for example to

announce the arrival of e-mail, notify you of machines crashing or computation jobs completing.

gateways

PSYC even supports foreign uniform schemes within its protocol so that routing messages to gateways is not a problem. Protocols, which do not have the ability to reach the receiver location-independently could be taught to do so by gatewaying them into PSYC. I will name a few, that may or may not be familiar to you:

- The BITnet MSG and its internet equivalent MSEND (RFC1312)
- The ESMTP commands SAML and SOML (send as mail or terminal, RFC821)

And since mobile telephony isn't based on internet protocols yet, gateways to GSMs short message service (SMS) would be fruitful. So far only gateways to e-mail have been implemented, which isn't suited for the purpose.

suggestions?

An internet messaging protocol is useful for a whole range of applications. Which ones come to *your* mind?

Other approaches to this particular issue

Dynamic DNS (exposes personal computer)
IPv6 (waiting for godot)
SIP (although beyond original scope)
Jabber™ (a bit cumbersome, but works)

Conferencing: rooms and groups

psyc://psyc.ai.mit.edu /@gnu.announce/

This is what a UNI of a conference session would look like. It points to a group manager object and suggests the client should ask the named server for access to this group, but it could actually call on all kinds of methods in this object. The "@" sign simply declares this object to be compliant to the room interface, so it can be entered.

Both conference control and routing are programmable in the manager of the room!

And yes, you can have more than one manager if you think one alone isn't safe.

Routing

The multicast principle is fundamental to PSYC

PSYC has a notion of routing and delivering to multiple recipients on the lowest layers of its

design. This sounds so obvious, it is hard to believe only very few multicast technologies really exist. Even the word itself has turned into a synonym for one particular technology, IP Multicast, which for most purposes isn't even suitable. With its basic concepts of context, logical targets and packet ids PSYC consumes a lot less bandwidth, makes it easier to detect abuse and implement efficient protection against "SPIM" (messaging SPAM).

junction networks

Let's take the junction networks as an example. They are one of several multicast strategies of PSYC. They are an IRC-like tree structure, but a lot more efficient since the involved rooms create an optimal tree themselves rather than relying on the topology of an IRC network. They are wonderfully suited for the delivery of news headlines, a lot more efficiently than unicast poll technlogies like RSS ever could be. Also applications like BitTorrent could make extensive use of the realtimeness of PSYC as an announcement medium.

context slaves

The context slaves are a much more everyday approach to multicast where better routing is automatically figured out as soon as there are more recipients linking into a transmission from a specific source. This happens behind your back and keeps traffic low.

multipeer to peer

You may not want your servers to deal with bandwidth intensive applications like file transfers, but rather have the users communicate with each other directly, peer to peer. That's very welcome in PSYC too and thanks to the UNI and UNL easier to set up. Then again you may find it interesting to create hybrid applications between backbone and peer to peer networking.

future multicasting

PSYC is conceptionally open to supporting new and even external multicast strategies. Did you know GPRS has builtin multicast subprotocols? We'd like to use those. Or even plain IP Multicast, should we ever need that. With packet ids we are able to support redundant topologies and/or lossy subprotocols, where it is fundamental to sort out duplicates, or where the duplicates tell which one is the fastest route.

Programmable conference control

programmable conference control

This doesn't mean the manager object will

always let clients in. In fact it can use whatever strategy it likes to, to determine who's allowed to enter the room.

You could let people in by asking for a password, or only those who have been invited or those who are in a fixed list of authenticated users, or maybe just everyone within the domain `*.fi`. The manager could even delay its decision to prompt humans for a decision, yes even ask the present members of the group to vote on the issue.

personalized rooms

I particularely like the thought of end users being able to set up their own persistent rooms just like they want them, by using HTML forms for instance. And should a feature be missing in the form, the user could ask the server admin to implement it.

bullet-proof chatrooms

In general only the manager object has the authority over the conference such that it can set up its properties and modify its member list. This protects conference sessions from take-overs as they happen on IRC.

distributed communication however

The actual message interchange between group members can happen directly peer to peer, through servers or proxies if you want it to, or via a multicast layer. This is achieved via PSYC's minimalistic conference control module, which basically gives the members a list of the other members, how they can be reached and, for the purpose of making moderated rooms, wether they are speakers or just listeners.

Friendcasting

friendsnets, friendcasting

The social network of your friendships can be modeled as a room. This way your presence updates can be delivered efficiently using multicasting. It has become extremely popular to send an email to all friends. By using friendcast you can also reach out for the friends of your friends, if you want to and they agree. So you could invite your entire social network to a party. Yes social networks, decentralized, open source and combined with proper multicasting.. There's quite some interesting potential in there!

Switching to other protocols

choice of protocols

The action of forwarding a person identification (UNI) to the current client software of this person isn't necessarily limited to one single UNL. A whole collection of protocol schemes, port numbers and even multicast schemes may be passed on so that more sophisticated communication systems may be integrated.

VoIP, SIP, Audio/Video-Conferencing

Now if we had UNL-schemes for all sorts of audio and video conference systems, PSYC would be able to negotiate for them easily. PSYC even provides much more evolved conference control than these systems usually provide. On the other hand developers of such real-time systems would no longer have to worry about conference control and simply concentrate on making good streaming protocols.

multicast routing with or without PSYC

It's useful to have a choice of protocols even for text-based communication. Multicast strategies, from the PSYC-based network of proxy servers to PSYC over Multicast IP (with the help of PIM-SM probably) up to totally independent multicast or even broadcast schemes.

Optional protocol modules

multimedia data

For completeness PSYC itself offers the ability of defining the content type contained in messages, so it doesn't necessarily have to be plain text, it may just as well be text/html or image/jpeg. It is then up to the implementor to decide what to do with unexpected content.

module for binary content

Appropriately, PSYC has the option of binary content delivery. Combined with a fragmentation option you can implement multicast file transfer over any of the supported multicast protocols, even hybrids. Or you might switch to a multiplexing variant of PSYC.

the concept of modules

But it is only necessary to implement those modules that are needed for the application, so a minimal PSYC server is quickly coded. The basic functionality of PSYC, delivery of text messages, is guaranteed however and per se very useful. You're welcome to conceive your own modules and add them to the negotiation mechanism.

Example of a protocol message (verbose)

example of a message in PSYC's unabbreviated format

```
=_identification psyc://psyc.cool.org/~cool
:_target      ~suzie

=_nickname    Coolman
=_nickname_alias Cool
=_description As cool as ice ice baby
:_action      whispers to you
_conversation
Hi! Any plans tonight?
.
```

And here's what Suzie is probably going to see:

here's what Suzie is probably going to see:

```
Coolman whispers to you: Hi! Any plans tonight?
```

Example of a protocol message (compact)

The same message in PSYC's compact format

```
=i    psyc://psyc.cool.org/~cool
:t    ~suzie

=n    Coolman
=n_alias      Cool
=D    As cool as ice ice baby
:a    whispers to you
c
Hi! Any plans tonight?
.
```

PSYC 1.0: compact keywords will be introduced.

The detailed specification of the protocols resides at http://www.psyc.eu/tech.en.html.

Ways in and out of IRC

Access by using an IRC client

The psyced software has the built-in ability to emulate an IRC server so you can access most of the strengths of the PSYC dimension with your favourite IRC client, like these:

- identity protected by password
- programmable answering machine services
- programmable configurable secure chatrooms

Gateways into IRC networks

IRC is still very popular, and since IRC cannot handle all its users within a single network a broad variety of networks has arisen since the dramatic split of 1990. That was not the intention of IRC. psyced provides gateway solutions which makes IRC users addressable from PSYC using irc:-URLs. You can't always be sure that the same

person answers on the other side, but it's better than nothing. PSYC could even act as gateway between IRC networks and the Jabber™.

More..

Compression and encryption

The protocol supports multiple encoding formats applied onto each other, like TLS/SSL on top of zlib. GnuPG/PGP would be an obvious option for end-to-end cryptography applications, but we prefer the OTR approach and would like to support that natively.

Future: interface descriptions for PSYC objects

For chat and messaging purposes the interface descriptions suggested by the ~ or @ characters in a UNI are sufficient. Ulterior methods may be *guessed*. Should developments go far beyond this, we may need an interface description syntax at some point. So far it hasn't happened.

Conceptional merge of chatrooms and buddy lists

Broadcasting your presence information to your friends is implemented as a special case of a chatroom where you have the chance to apply all features and flexibility of elaborate conference control to manage your contacts, and at the same time we end up with less code.

What's available today?

psyced Daemon

psyced isn't just a PSYC server; it also implements the functionality of PSYC clients allowing users of Java applets, Telnet, Jabber™ and IRC clients to enter the PSYCspace. Additionally psyced can communicate with Jabber™ servers and provide chatrooms for PSYCers, IRCers and Jabberers all at once. It also provides gateways to several IM- and IRC-networks as well as serving as a web server for distributed multi-user applications. It is implemented in LPC and released as open source. Scalable, programmable, effective.

Clients

... are not so important at the current stage of PSYC development, since psyced acts as client simulator and allows you to use your favorite IRC or jabber client applications. You can even just use the web access if you're not planning to install your own psyced. The developers themselves currently like to use IRC clients,

PsycZilla, psycion or an enhanced unix telnet software called powwow.

Once you have become familiar using psyced as a server, you may want to dig deeper and look at native PSYC applications:

PsycZilla

Based on Mozilla you can either install it as a Firefox Extension or standalone. It is a graphical crossplatform client with neat web integration. Absolutely try it out! PsycZilla!

psycion

psycion is the most elaborate PSYC client to date. It is aimed at unix console users and comes with a very fancy curses-based multi-screen interface. The code is modular so it is easy to add a GUI. psycion is being distributed with perlpsyc.

perlpsyc

Net::PSYC comes in a package with a collection of PSYC automations, command line messaging scripts, the cool psycion client and even an MP3 player you can remote control using PSYC messages or the command 'psyccmd <command>' from any shell prompt.

jaPSYC, Psychedelic and other Java™ apps

Mario 'BitKoenig' Holbe started out developing this library in Java back in 1996 while I was sorting out basics of the protocol syntax. Later on, I decided to motivate him to finish his work. The result is an amazingly elaborate implementation of the protocol suited for all sorts of client, server or hybrid applications. This development library is available free of charge as open source. about:Java

About

Carl von Loesch has been developing chat systems since 1988. During his work on IRC software he introduced the notorious **/me** command. Commercially successful with PSYC-based high scalability chat solutions from symlynX. Join our upcoming MTV Europe Music Awards Chat Event! And because all of this is actually quite boring he also makes music. You can buy it on vinyl or CD. And you can listen to his radio shows.

http://about.psyc.eu