

COMP208 - Group Software Project

Ballmer Peak

M. Chadwick; Choi, S.F; P. Duff; L. Prince; A.Senin; L. Thomas

March 20, 2014

Contents

I	Design	6
1	Proposal Summary	7
2	Architecture	8
2.1	Network Architecture	8
2.2	System Architecture	9
2.3	Data Flow Diagram	9
3	Use Case Diagrams	11
4	Protocol	17
4.1	High Level Summary of Protocol	17
4.2	Client-Server Protocol	18
4.3	Client-Client Protocol	19
4.4	Summary	19
4.5	Message Formatting	19
4.5.1	Unencrypted Messages	19
4.5.2	Encrypted Messages	20
4.6	Claiming a Username	20
4.7	Revoking a Key	20
4.8	Profile Data	20
4.9	Inter-User Realtime Chat	21
4.10	Posting to own wall	21
4.11	Posting on another users wall	22
4.12	Commenting	22
4.13	Liking	22
4.14	Events	22

<i>CONTENTS</i>	<i>3</i>
5 Class Interfaces	24
5.1 Class Interfaces	24
5.2 Class Diagram	29
6 Pseudocode	30
6.1 Server	30
6.2 Client	30
6.3 Crypto	31
6.4 Database	32
6.5 Network Connection	33
6.6 Parser	34
7 Database	36
7.1 Database design description	36
7.1.1 user table	36
7.1.2 user, is_in_category, category table	36
7.1.3 user, is_invited, events table	36
7.1.4 user, allowed_to, wall_post table	37
7.1.5 user, has_like, wall_post table	37
7.1.6 user, has_like, has_comment table	37
7.1.7 user, has_comment, wall_post table	37
7.1.8 user, has_comment table	37
7.1.9 user, is_in_message, private_message table	38
7.1.10 message_claim table	38
7.1.11 key_revoke table	38
7.1.12 login_logout_log table	38
7.2 Table layout of the database	39
8 Transaction details	43
9 User Interfaces	45
9.1 Interface Research	45
9.1.1 Swing	45
9.1.2 Abstract Window Toolkit	46
9.1.3 Standard Widget Toolkit	46
9.1.4 GWT	47
9.1.5 Javascript	47
9.2 GUI Design	48

9.2.1	Client Design	48
9.2.2	Server Design	50
9.3	Future Work	51
10	Business Rules	54
11	Gantt Chart	55
12	Glossary	59
Appendices		
A	Deadlines	62
B	Licence	63
B.1	Statement of Purpose	63
B.2	Copyright and Related Rights	64
B.3	Waiver	64
B.4	Public License Fallback	65
B.5	Limitations and Disclaimers	65
B.6	Included Works	66
B.7	jquery.js Licence	66
B.8	todonotes.sty licence	67
B.9	The LaTeX Project Public License	67
B.9.1	PREAMBLE	67
B.9.2	DEFINITIONS	67
B.9.3	CONDITIONS ON DISTRIBUTION AND MODIFICATION	68
B.9.4	NO WARRANTY	70
B.9.5	MAINTENANCE OF THE WORK	71
B.9.6	WHETHER AND HOW TO DISTRIBUTE WORKS UNDER THIS LICENSE	72
B.10	CC0 licence and licence.png licence	74
B.11	Creative Commons Attribution 4.0 International Public License	74
B.11.1	Section 1 - Definitions.	75
B.11.2	Section 2 - Scope.	76
B.11.3	Section 3 - License Conditions.	77
B.11.4	Section 4 - Sui Generis Database Rights.	78
B.11.5	Section 5 - Disclaimer of Warranties and Limitation of Liability.	78
B.11.6	Section 6 - Term and Termination.	79
B.11.7	Section 7 - Other Terms and Conditions.	79

B.11.8 Section 8 - Interpretation.	79
B.12 ulem.sty licence	80
B.13 turtles.ttf Licence	80
B.14 tikz-uml.sty licence	80
B.15 tikz-styles.sty licence	80

Part I

Design

Chapter 1

Proposal Summary

The project, a security based social media network, will have multiple components to be investigated and used in this design section. The key critical components to be looked at consist of:

- Database
- Client
- Client GUI
- Server
- Server GUI
- Mobile GUI (future work)

Of each of these components we should look at how they will impact their respective uses in order to best make use of their full functionality. We will look at multiple possible and practical solutions for the above criteria, making sure the best solution is chosen. We will also look at possible work in the future, or any areas to continue with into the coming stages.

The requirements section has helped so far through analysis of existing social media networks and how they have implemented their networks, along with how their interfaces react to the user.

Chapter 2

Architecture

2.1 Network Architecture

Turtlenet is a centralized service, whereby a large number of clients connect to a single server which provides storage, and facilitates communication between clients.

Due to the inherently limited network size (5-50K users per server depending on percentage of active participants vs consumers and local internet speeds) we recommend that servers serve a particular interest group or geographic locality.

Clients send messages to, and only to, these central servers. Due to the fact that all messages (except CLAIM messages, see client-server/client-client protocols for details) are encrypted the server does not maintain a database, it cannot; rather clients each maintain their own local database, populated with such information to which they have been granted access.

When a client wishes to send a message to a person, they encrypt the message with the public key of the recipient¹ and upload it to the server. It is important to note that all network connections are performed via Tor.

When a client wishes to view messages sent to them, they download all messages posted to the server since they last downloaded all messages from it, and attempt to decrypt them all with their private key; those messages the client successfully decrypts (message decryption/integrity is verified via SHA256 hash) were intended for it and parsed. During the parsing of a message the sender is determined by seeing which known public key can verify the RSA signature.

Due to the nature of data storage in client-local databases, all events and data within the system must be represented within these plaintext messages. This is achieved by having multiple types of messages (see client-client protocol).

¹using RSA/AES, see protocol for details

2.2 System Architecture

The system has a number of modules which interact with one another via strictly defined interfaces. Each module has one function, and interacts as little as possible with the rest of the system. The modules and their interactions are shown below. NB: $a \rightarrow b$ denotes that data passes from module a to module b , and $a \leftrightarrow b$ similarly denotes that data passes both from a to b and from b to a .

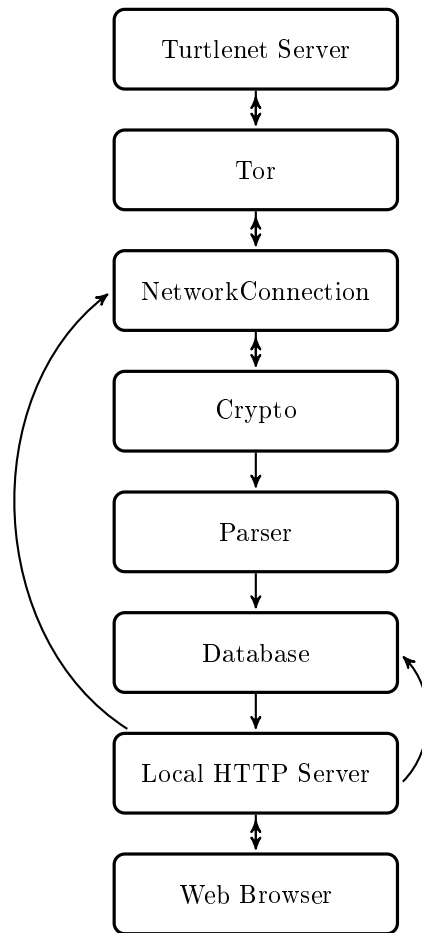


Figure 2.1: Module Interaction

2.3 Data Flow Diagram

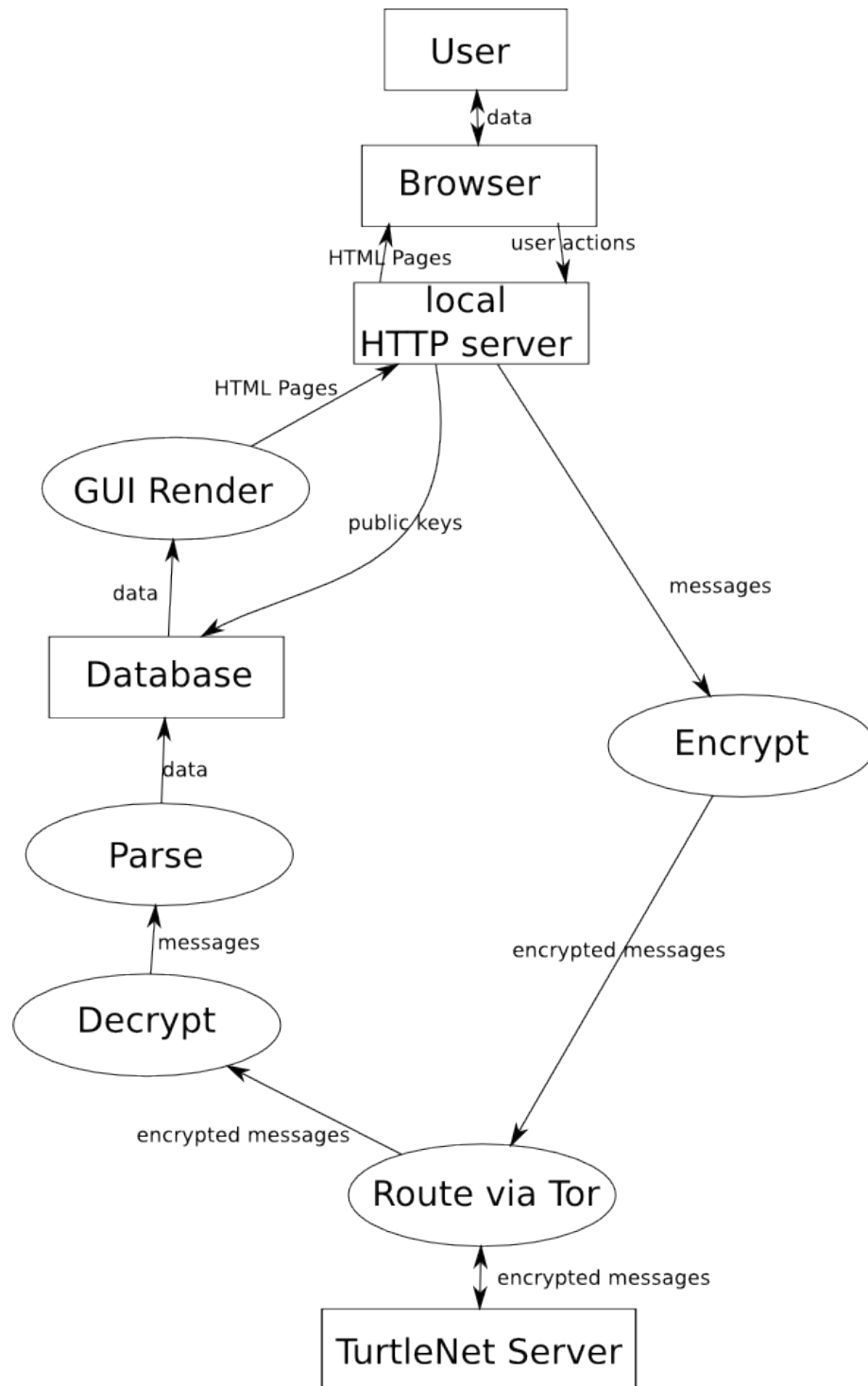
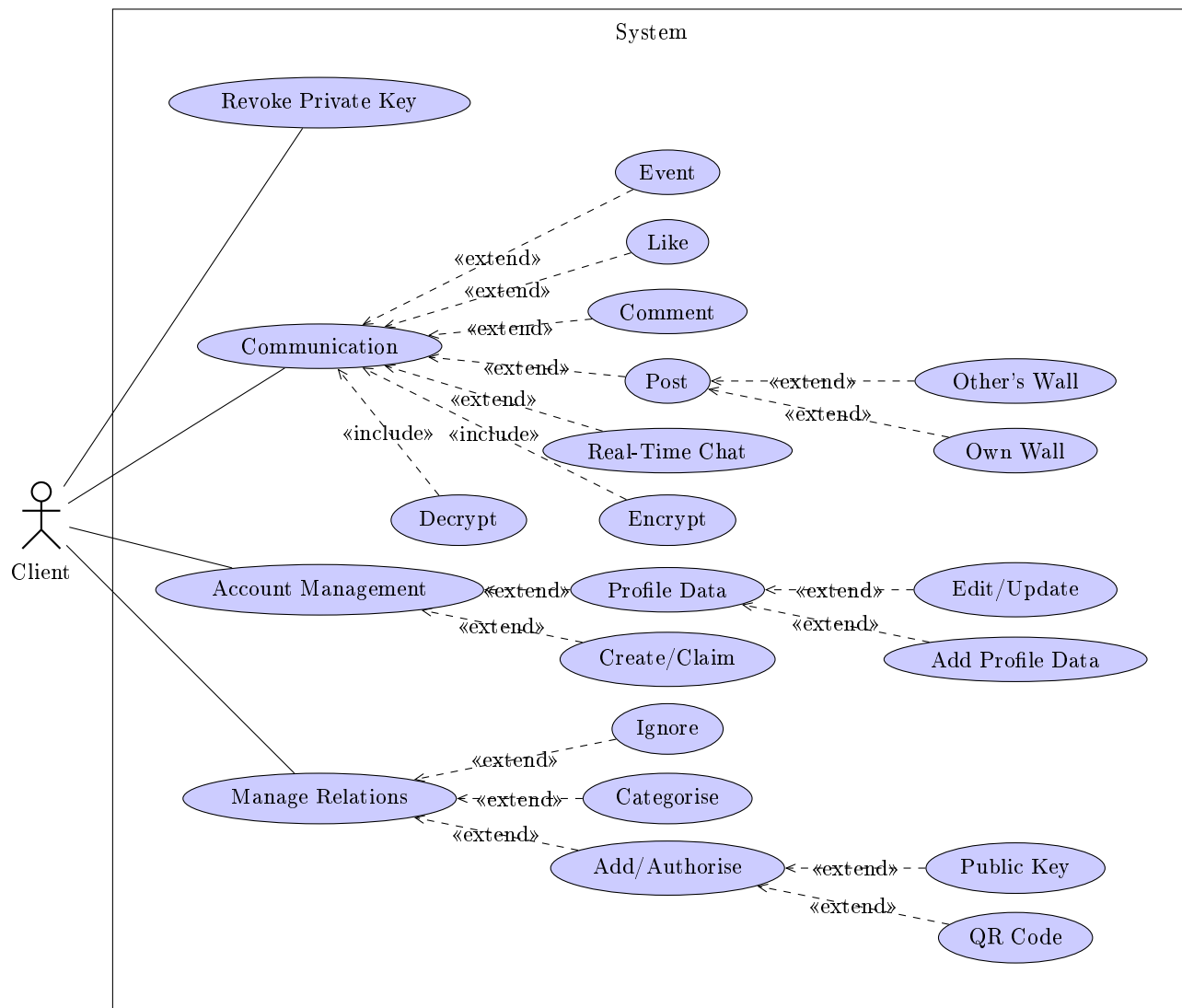


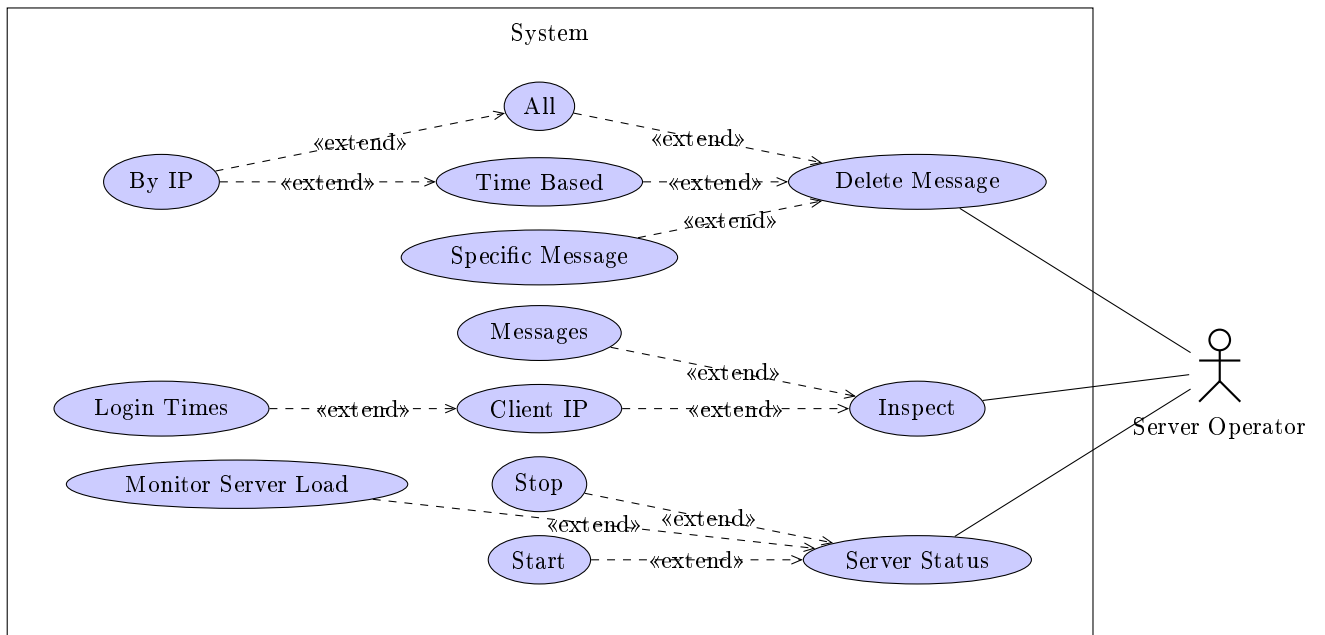
Figure 2.2: Data Flow Diagram

Chapter 3

Use Case Diagrams

Here we have a use case diagram displaying an actors interaction with our system. It shows the functionality available to both the client, and the operator. We also have a sequence diagram to augment the use case diagrams.





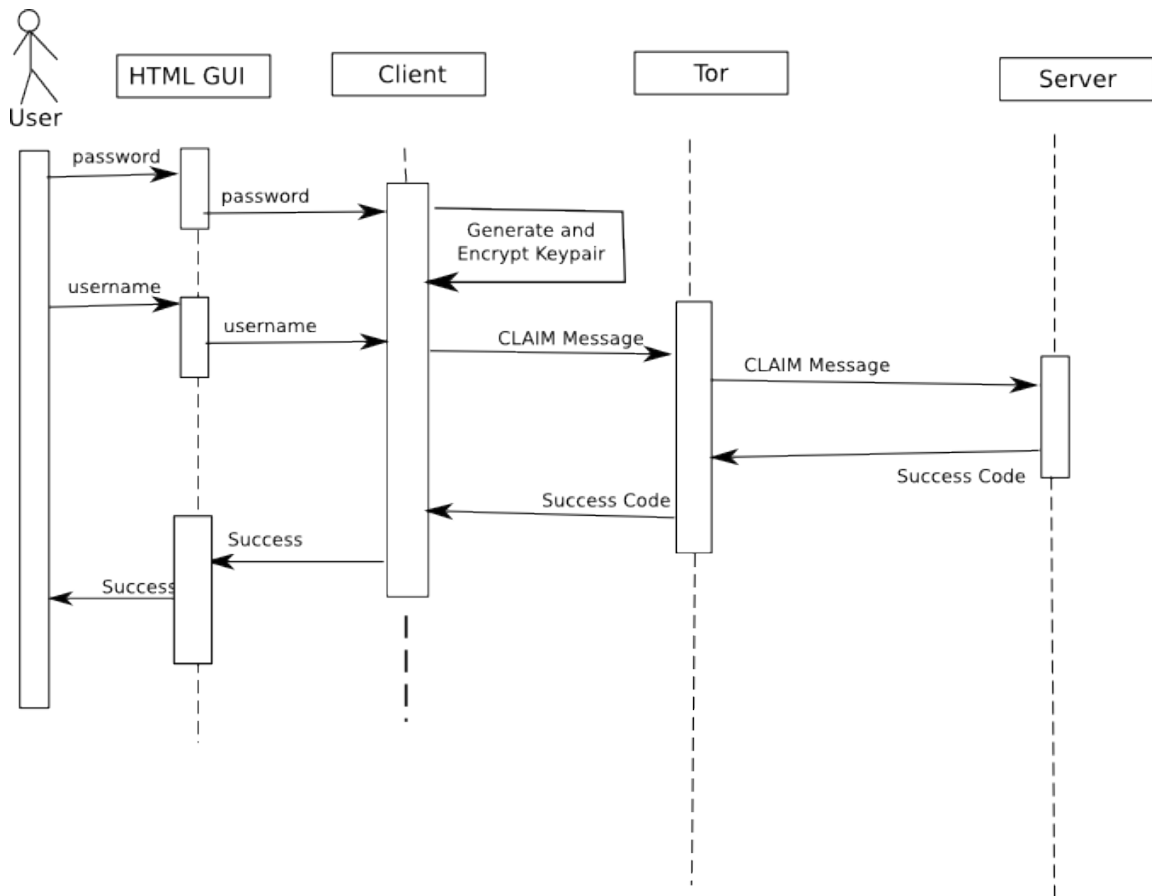


Figure 3.1: Sequence Diagram - Registering

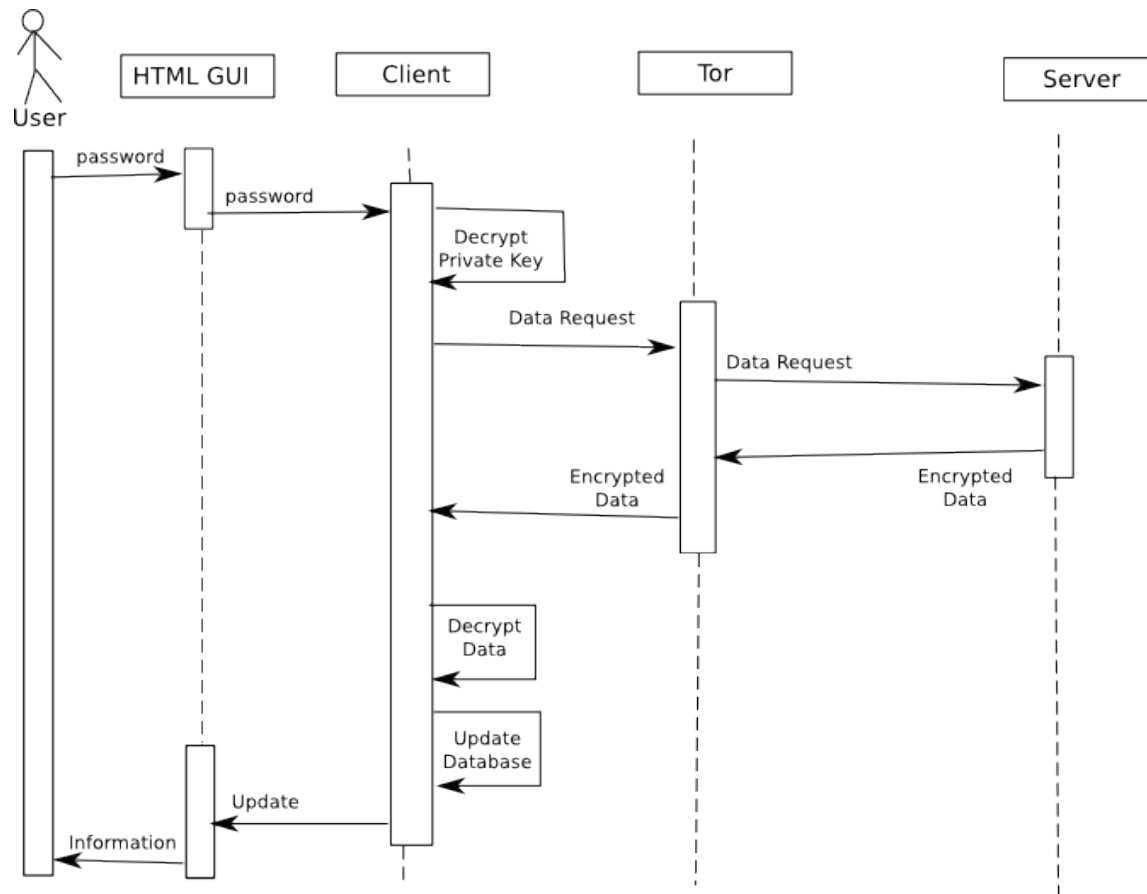


Figure 3.2: Sequence Diagram - Recieving Data

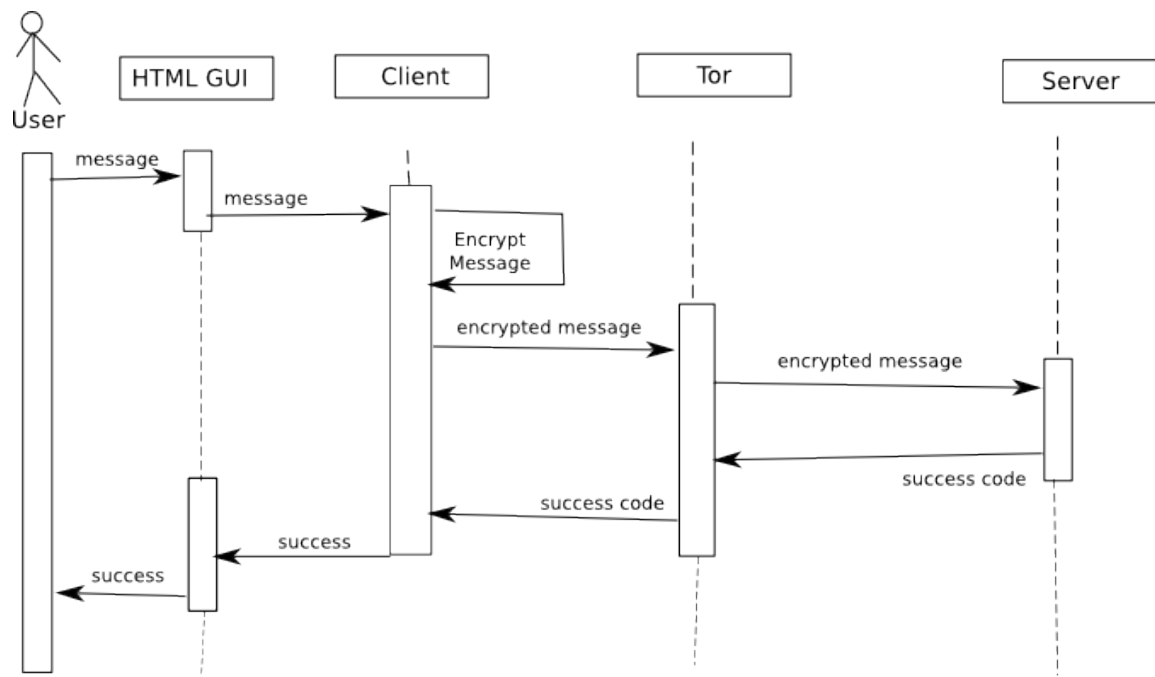


Figure 3.3: Sequence Diagram - Sending Data

Chapter 4

Protocol

4.1 High Level Summary of Protocol

Creating an account is done by generating an RSA keypair, and choosing a name. An unencrypted (but signed) message is then posted to the server associating that keypair with that name. In this way, by knowing the public key of someone, you may discover their name in the service, but not vice versa.

Connecting for the first time Every unencrypted message stored on the server is downloaded (signed nicknames and nothing more). At this time the local database contains only signed messages claiming usernames. The public keys are not provided, these are of use only when you learn the public key behind a name. The rationale for not providing public keys is provided in the section regarding adding a friend. Messages posted after your name was claimed will require downloading too, as once you claim a name people may send you messages. It's worth noting that messages from before you connected for the first time are now downloaded because they can not have been sent to you (with a compliment client) if someone retroactively grants you permission to view something they publish it as a new message with an old timestamp; the sole exception to this is when you connect using a new device, in which case all messages since you first claimed a name will be downloaded.

Connecting subsequently The client requests every message stored on the server since the last time they connected up to the present. Decryptable messages are used to update the local DB, others are discarded.

Continued connection During a session the client requests updates from the server every

0.5-5 seconds (configurable by the user).

Adding a friend is performed by having a friend email (or otherwise transfer) you their public key. This is input to the client, and it finds their username (via public posting that occurred when registering). You may now interact with that person. They may not interact with you until they receive your public key. Public key transferral will be performed via exchanging plaintext base64 encoded strings, or QR codes. The user will be prompted, after retrieving the username of the user, to categorise them.

Talking with a friend or posting on your wall is achieved by writing a message, signing it with your private key, and encrypting one copy of it with each of the recipients public keys before posting it to the server. The client prevents one from posting a message to someone's public key if they have not claimed a nickname.

Posting to a friends wall, commenting and liking may be requested by sending a EPOST/COMMENT/LIKE message to the friend (upon whose wall/post you are posting, commenting or liking), when that friend logs in they will receive your request, and may confirm or deny it. If they confirm then they take your (signed) message and transmit it to each of their friends as previously described. Given that authentication is entirely based on crypto signatures it doesn't matter that your friend relays the message. This is required because it is impossible for one to know who is able to see the persons wall, post, or comment upon which you seek to post, like, or comment.

4.2 Client-Server Protocol

The client-server architecture is necessarily simple.

The client connects to the server, sends a single command, receives the servers response and then disconnects. The following shows commands sent by the client, and the servers action in response.

command	purpose	servers action
t	get the server time	sends back the current time (unix time in milliseconds)
s <i>utf-8_text</i>	send messages	the text sent is stored on the server
get <i>ms_unix_time</i>	get new messages	every message stored since the given time is sent
c <i>utf-8_text</i>	claim a username	the text sent is stored on the server, with a special filename

Table 4.1: Client-Server Protocol

Every command is terminated with a linefeed. Every response from the server will be terminated with a linefeed. The last line sent by the server will always be "s" for success, or "e" for failure (this is omitted from the above table).

CLAIM messages (sent with `c`) will be parsed by the `Message` class and the username extracted for use in a filename. The filename of claim messages is as follows `<unix_time_in_ms>_<username>`; the filename of all other messages is as follows `<unix_time_in_ms>_<SHA256_hash>`.

4.3 Client-Client Protocol

4.4 Summary

All client-client communication is mediated by the server. When one client wishes to send a message to another it encrypts the message with the public key associated with the recipient and uploads it to the server. When one client wishes to receive a message it downloads all new messages from the server and parses those it can decrypt. This is performed in order to hide who receives a message. All messages except CLAIM messages are encrypted. Multiple recipients imply multiple messages being uploaded, this is taken for granted in the text which follows.

4.5 Message Formatting

4.5.1 Unencrypted Messages

Messages have a command (or type), which specifies the nature of the message; messages have content, which specifies the details of the message; messages have an RSA signature, which authenticates the message; messages have a timestamp, which dates the message down to the millisecond, the time format is unix time in milliseconds.

Messages are represented external to the system as utf-8 strings, and internally via the `Message` class. The string representation is as follows:

`<command>\<signature>\<content>\<timestamp>`

Backslashes are literal, angle brackets denote placeholder values where data specific to a message is placed.

An example follows:

`POST\<signature>\Hello, World!\1393407435547`

backslashes in message content are escaped with another backslash, signatures are base64 encoded SHA256/RSA signatures of the content of the message concatenated with a decimal string representation of the timestamp. All text is encoded in UTF-8.

4.5.2 Encrypted Messages

Encrypted messages contain the AES IV's; the RSA encrypted AES key; and the AES encrypted message.

Messages are encrypted by encoding the entire message to be sent with UTF-8; encrypting the message with a randomly generated AES key; encrypting the AES key with RSA; encoding the RSA encrypted AES key in base64; encoding the (random) AES initialization vectors in base64 and concatenating these three parts with a backslash between each. The format follows:

$$\langle AES\ IV \rangle \backslash \langle RSA\ encrypted\ random\ AES\ key \rangle \backslash \langle AES\ encrypted\ message \rangle$$

Backslashes are literal, angle brackets denote placeholder values where data specific to a message is placed.

4.6 Claiming a Username

Each user (keypair) should claim one username. Uniqueness is enforced by the server, and so not relied upon at all. Usernames are useful because public keys are not human readable. In order to claim a username, one must send an unencrypted CLAIM message to the server. The format follows:

$$CLAIM \backslash \langle signature \rangle \backslash \langle username \rangle \backslash \langle timestamp \rangle$$

4.7 Revoking a Key

If a users private key should be leaked, then they must be able to revoke that key. This is done by sending a REVOKE message to the server. All content signed by the private key after the stated time will be flagged as untrusted. The format follows:

$$REVOKE \backslash \langle signature \rangle \backslash \langle time \rangle \backslash \langle timestamp \rangle$$

4.8 Profile Data

Users may wish to share personal details with certain people, they may share this information via profile data. Profile data is shared using PDATA messages. A PDATA message contains a list of fields, followed by a colon, followed by the value, followed by a semicolon. The format follows:

$$PDATA \backslash \langle signature \rangle \backslash \langle values \rangle \backslash \langle timestamp \rangle$$

The format for values follows:

$\langle field \rangle: \langle value \rangle; \dots$

An example follows:

PDATA\<signature>\name:Luke Thomas;dob:1994;\<timestamp>

4.9 Inter-User Realtime Chat

Users can chat in real time, this is achieved by sending a CHAT message to all people you wish to include in the conversation. This message includes a full list of colon delimited public keys involved in the chat. The format follows:

CHAT\<signature>\<keys>\<timestamp>

The format for keys follows:

$\langle key \rangle: \langle another_key \rangle \dots$

An example follows:

CHAT\<signature>\<key1>:<key2>\<timestamp>

Following the establishment of a conversation, messages may be added to it with PCHAT messages, the format follows:

PCHAT\<signature>\<conversation>:<message>\<timestamp>

Whereby $\langle conversation \rangle$ denotes the signature present on the establishing message. An example follows:

PCHAT\<signature>\9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08:First!\<timestamp>

4.10 Posting to own wall

When a user posts to their own wall they upload a POST message to the server of the following format.

POST\<signature>\<message>\<timestamp>

The format of message is merely UTF-8 text, with backslashes escaped with backslashes.

An example follows which contains the text "Hello, World!", a newline, "foo \bar\baz":

POST\<signature>\Hello, World!
foo\\bar\\baz\<timestamp>

4.11 Posting on another users wall

A user may request to post on a friends wall by sending them an FPOST message, the poster may not decide who is able to view the message. The format is identical to that of a POST message, except for the command and singular recipient. An example follows:

```
FPOST\<signature>\Hello, World!\<timestamp>
```

Upon receipt of an FPOST message the friend is prompted by the client to choose whether or not to display it, and if so who may view it. Once this is done the friend reposts the message with the command changed to POST instead of FPOST as they would post anything to their own wall. This works because authentication is entirely based on RSA signatures so in copying the original signature the friend may post as the original author provided they don't alter the message (and thus its hash and required signature).

4.12 Commenting

Commenting works similarly to posting on another's wall, so an explanation of details of how it occurs is not provided (see prior section). The only difference is the format of a CMNT message from an FPOST message. The format of a CMNT message is as follows:

```
CMNT\<signature>\<hash>: <comment>\<timestamp>
```

Where *<hash>* denotes the hash of the post or comment being commented upon. An example comment follows:

```
CMNT\<signature>\v/sXfb3DG2qT2k2hXIH4csJy1yEG+TANRbbxQw1VkSE=: Yeah, well,  
that's just like, your opinion, man.\<timestamp>
```

4.13 Liking

Like messages are identical to comments except for the command and the the fact that no "*<comment>*" follows the hash. An example like follows:

```
LIKE\<signature>\v/sXfb3DG2qT2k2hXIH4csJy1yEG+TANRbbxQw1VkSE=\<timestamp>
```

4.14 Events

A user may have the client remind him of an event by alerting him when it occurs. A user may inform others of events, and they may choose to be reminded about them. When a user creates

an event just for themselves they just create a normal event and only inform themselves of it. An event is created by posting an EVNT message to the server. The format follows:

EVNT*<signature>*\<event_start_time>: <event_end_time>: <event_name>\<timestamp>

An example follows of a reminder for bobs birthday which occurs on the 14th of January, the event was created on the second of January:

EVNT*<signature>*\1389657600000: 1389744000000: bobs birthday\1388676821000

Chapter 5

Class Interfaces

5.1 Class Interfaces

The following is a description of the public functions of all public classes. Many classes have inner private classes they use for convenience, however to simplify interaction between parts of our system ('modules') we have very few convenience classes.

return	function	description
void	main()	(static) starts the server

Table 5.1: Server

return	function	description
void	main()	(static) constructs and starts all necessary classes and threads, runs the main loop

Table 5.2: Client

return	function	description
N/A	NetworkConnection()	Constructs a NetworkConnection and connects to the given URL (through tor)
void	run()	periodically download new messages until asked to close, downloaded messages are stored in a FIFO buffer
void	close()	kills the thread started by run()
boolean	hasMessage()	return true if there is a message in the buffer, false otherwise
String	getMessage()	return the oldest message in the buffer
boolean	claimName()	claim a given username, returns true on success, false otherwise
void	revokeKeypair()	revokes your keypair
void	pdata()	adds or updates profile information
void	chat()	begins or continues a conversation
void	post()	post a message to your wall
void	fpost()	post a message to a friends wall
void	comment()	comment on a comment or post
void	like()	like a comment or post
void	event()	create an event

Table 5.3: NetworkConnection

return	function	description
boolean	keysExist()	(static) return true if the user has a keypair, false otherwise
void	keyGen()	(static) generate a keypair for the user
PublicKey	getPublicKey()	(static) returns the users public key
PrivateKey	getPrivateKey()	(static) returns the users private key
String	sign()	(static) returns an RSA signature of the passed string
boolean	verifySig()	(static) returns true if author signed msg, false otherwise
String	encrypt()	(static) returns an encrypted message constructed from the passed parameters
Message	decrypt()	(static) decrypts the passed string, returns the appropriate message, on failure a NULL message is returned
String	base64Encode()	(static) base64 encodes the passed data, returns the string
byte[]	base64Decode()	(static) base64 decodes the passed data, returns the byte[]
String	encodeKey()	(static) encodes a public key as a string, returns that string (X509)
PublicKey	decodeKey()	(static) decodes a public key encoded as a string, returns that public key(X509)
String	hash ()	(static) returns the SHA256 hash the the passed string as a hex string
int	rand ()	(static) returns a pseudorandom value \leq max and \geq min

Table 5.4: Crypto

return	function	description
void	parse()	(static) parses a sting message, records parsed data in the database

Table 5.5: Parser

return	function	description
void	addClaim()	adds a username CLAIM message
pair<string,string>[]	getClaims()	gets all CLAIMs to usernames
string[]	getUsernames()	gets all usernames
void	addRevocation()	adds a keypair revocation
pair<PublicKey, long>[]	getRevocations()	gets all revocations
boolean	isRevoked()	returns the time a key was revoked, if the given key has not been revoked then 0 is returned.
void	addPData()	adds (or amends existing) profile data
string	getPData()	gets the specified piece of profile data for a specified user
void	createChat()	creates new chat
pair<string,string>[]	getChat()	returns messages from a given chat
void	addToChat()	adds a post to a given chat
void	addPost()	creates new post, on your or another's wall
pair<string,string>[]	getPosts()	gets all posts either within timeframe, or from certain people within a timeframe
void	addComment()	adds a comment onto post or comment
pair<string,string>[]	getComments()	gets all comments for a post or comment
void	addLike()	likes a post or comment
String[]	getLikes()	gets all likes from certain person within a timeframe
int	countLikes()	gets the number of likes for a comment or post
void	addEvent()	adds new event
pair<string,long>[]	getEvent()	gets all events within timeframe
void	acceptEvent()	accepts notification of an event
void	declineEvent()	declines notification of an event
void	addKey()	adds a public key to the DB
PublicKey[]	getKey()	gets the public key for a username, or all which are stored
string	getName()	gets a username for the given public key
void	addCategory()	adds a new category to the DB
void	addToCategory()	adds a user to a category

Table 5.6: Database

return	function	description
N/A	GUI()	Constructs a GUI
void	run()	continually updates the GUI from the DB
void	close()	kills the GUIserver thread
boolean	isRunning()	returns true if the GUIserver is running, false otherwise

Table 5.7: GUI

return	function	description
N/A	Message()	Constructs a message with given data
Message	parse()	(static) parses the string representation of a message into a message
String	toString()	creates a string representation of the message
String	getCmd()	returns the type of message
String	getContent()	returns the content of the message
String	getSig()	returns the RSA signature on the message
long	getTimestamp()	returns the timestamp on the message

Table 5.8: Message

return	function	description
N/A	Pair()	Constructs a pair with given data
A	first()	returns the first value passed to the constructor
B	second()	returns the second value passed to the constructor

Table 5.9: Pair<A, B>

5.2 Class Diagram

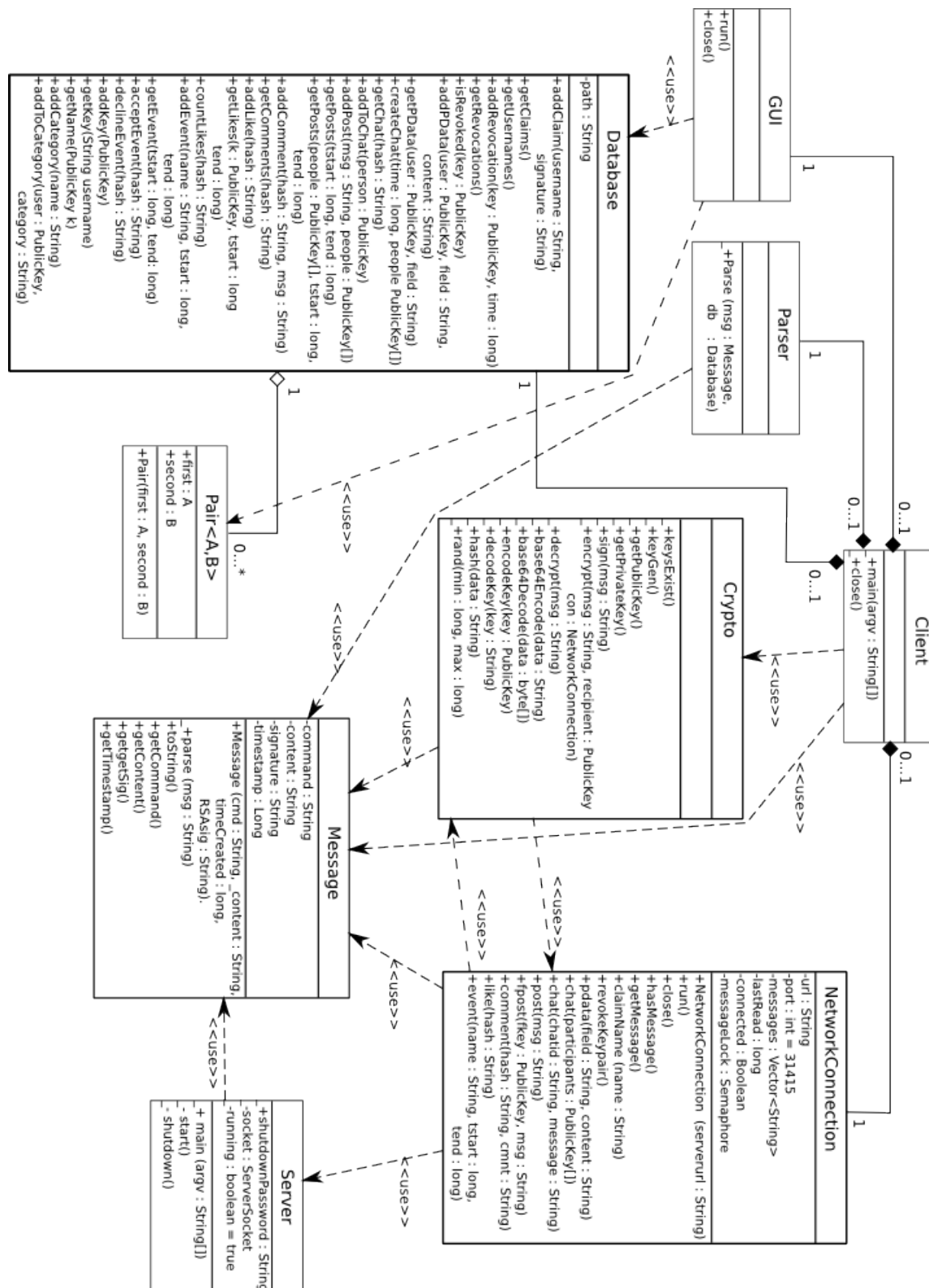


Figure 5.1: UML Class Diagram

Chapter 6

Pseudocode

6.1 Server

```
static void main () {
    startGUIthread()
    startServer()
}

static void start () {
    socket = new ServerSocket(port)
    while (running) {
        incoming = socket.accept()
        t = new Thread(new Session(incoming))
        t.start()
    }

    shutdown()
}
```

6.2 Client

```
static void main () {
    NetworkConnection connection = new NetworkConnection("server.tld")
    Thread networkThread = new Thread(connection)
    Database db = new Database("./db")
}
```

```

GUI                gui                = new GUI(db, connection)
Thread            guiThread          = new Thread(gui)

if (!Crypto.keysExist())
    Crypto.keyGen()

networkThread.start()
guiThread.start()

while (gui.isRunning())
    while (connection.hasMessage())
        Parser.parse(Crypto.decrypt(connection.getMessage()), db)
}

```

6.3 Crypto

```

keyGen () {
    keypair = generateRSAkeypair()
    pw      = GUI.getUserInputString()
    filesystem.write("keypair", Crypto.aes(pw, keypair))
}

static String sign (String msg) {
    byte[] sig = SHA1RSAsign(msg.getBytes("UTF-8"), Crypto.getPrivateKey())
    return Crypto.Base64Encode(sig)
}

static String encrypt(String cmd, String text, PublicKey recipient,
                      NetworkConnection connection) {
    Message msg = new Message(cmd, text, connection.getTime()+Crypto.rand(0,50),
                              Crypto.sign(text))

    //encrypt with random AES key with random initialization vectors
    byte[] iv = new byte[16]
    byte[] aeskey = new byte[16]

    fillWithRandomData(iv);
    fillWithRandomData(aeskey);
}

```

```

byte[] aesCipherText = aes(aeskey, iv, msg.toString().getBytes("UTF-8"))

//encrypt AES key with RSA
byte[] encryptedAESKey = rsa(Crypto.getPrivateKey(), aeskey)

// "iv\RSA encrypted AES key\cipher text"
return Base64Encode(iv) + "\\ " + Base64Encode(encryptedAESKey) +
    "\\ " + Base64Encode(aesCipherText)
}

static Message decrypt(String msg) {
    //handle claim messages (which are the only plaintext in the system)
    if (msg.substring(0,2).equals("c "))
        return Message.parse(Base64Decode(msg.substring(2)))

    //handle encrypted messages
    String[] tokens = new String[3]
    tokens = tokenize("msg", "\\")

    byte[] iv          = Base64Decode(tokens[0])
    byte[] cipheredKey  = Base64Decode(tokens[1])
    byte[] cipherText   = Base64Decode(tokens[2])

    //decrypt AES key
    byte[] aesKey = rsaDecrypt(cipheredKey, getPrivateKey())

    //decrypt AES Ciphertext
    aes.init(Cipher.DECRYPT_MODE, aesKeySpec, IVSpec)
    byte[] messagePlaintext = aesDecrypt(cipherText, aesKey, iv)

    return Message.parse(messagePlaintext)
}

```

6.4 Database

Most database functions are just going to construct parameterized SQL queries to be sent to the database from passed parameter values. The exceptions which include significant computing are

listed here:

```

void addKey (PublicKey k) {
    for each row r in table message_claim
        if (Crypto.verifySig(r.signature, k))
            addFriend(new Friend(k, r.username))
}

PublicKey[] getKey (String username) {
    PublicKey[] keys
    for each row r in table user
        if (r.username == username)
            keys.add(r.public_key)
    return keys
}

void addToCategory (Friend f, String category) {
    for each row r in table wall_post
        if (r.permission_to includes category)
            sendMessage(r, f)
}

```

6.5 Network Connection

The vasy majority of messages here merely construct the appropriate message from the parameters and pass it to `serverCmd()`

```

void main (String _url) {
    url = _url
    messages = new Vector<String>()
    messageLock = new Semaphore(1)
    connected = true

    File lastReadFile = new File("./db/lastread")
    lastRead = Long.parseLong(lastReadFile.readLine())
}

void run () {
    while(running) {

```

```

        sleep(delay)
        downloadNewMessages()
    }
}

String[] serverCmd(String cmd) {
    Socket s;
    BufferedReader in;
    PrintWriter out;

    //connect
    s = new Socket(new Proxy(Proxy.Type.SOCKS, new InetSocketAddress("localhost", 90
    s.connect(new InetSocketAddress(url, port))
    in = new BufferedReader(new InputStreamReader(s.getInputStream()))
    out = new PrintWriter(s.getOutputStream(), true)

    //send command
    out.println(cmd);
    out.flush();

    //recieve output of server
    Vector<String> output = new Vector<String>();
    String line = null;
    do {
        line = in.readLine();
        if (line != null)
            output.add(line);
    } while (line != null);
}

```

6.6 Parser

```

void parse (String msg, Database db) {
    Message m = Message.parse(msg)
    if (m.cmd == "PDATA") {
        String[] tokens = tokenize(msg.content, ":")
    }
}

```

```
        db.addPData(tokens[0], tokens[1])
    } else if (m.cmd == "REVOKE") {
        PublicKey key
        for row r in table users
            if Crypto.verifySig(r.public_key, m.signature)
                key = r.public_key
        db.addRevocation(key)
    } else if {
        etc ...
    }
```

Chapter 7

Database

7.1 Database design description

Note the difference between 'main user' and 'user'. Main user refers to the user who owns the local database. 'User' or 'other user' refers to other users, usually the relations of the main user.

7.1.1 user table

This table stores user details, which includes the main user's own details and its relations. As the user makes a new relation with another user, its details will be stored in this table. Every user has their own public key which uniquely identifies their accounts which also be stored in this table.

7.1.2 user, is_in_category, category table

With the category table, the user can create new categories to group his relations. As it is possible for many users to belong in many categories, the *is_in_category* table is needed to identify which set of users belong in the categories.

7.1.3 user, is_invited, events table

These tables suggest that users can create events. One particular feature regarding these tables that on the *is_invited table*, where the user (the main one) can invite anyone individually from the relations list or as a group from the category list. However, there will be no tuples added under this table when another user posts the event. Reason being is that the main user is not allowed to see who the list of other users invited in the event which was not created by the main user.

When the main user creates an event, he invites other people, either from the user table or from the category table or both. Once the invitation is sent out to those users, the users can either accept or reject the invitation. Using the *decision* attribute from the *is_invited* table, if decision has not been made, it will be NULL. If user accepts the invitation, it will be 1 for true. If rejected, it will be 0 for false.

7.1.4 user, allowed_to, wall_post table

When users create post, its data will be inserted into the *wall_post* table. The attribute *from* refers to the user who has created the post, whilst the attribute *to* refers to the user who is referred or mentioned in this post. The main user can also choose to allow a set of his relations to view his post. Using the *allowed_to* table, similar as the *is_invited* table, the main user can select his relations either individually or through categories or both. If the post is created by another user, no tuples will be inserted into the *allowed_to* table.

7.1.5 user, has_like, wall_post table

Users can like any posts that appears in his main wall or personal wall. When a post is liked, a new tuple is created in the *has_like* table to identify who liked the post, which post is liked, and the time the post is liked. These likes are counted and displayed in the GUI showing how many users have liked this post.

7.1.6 user, has_like, has_comment table

Other than liking posts, users can like individual comments as well. Same feature as liking the post by this time, data is inserted into the attribute *comment_id* from the *has_like* table to show which particular comment has been liked by this user.

7.1.7 user, has_comment, wall_post table

Users can comment on posts. When post is commented on, a new tuple will be added into the *has_comment* table on information like the content of the comment, which post has been commented on, who commented on the post, and the time of comment.

7.1.8 user, has_comment table

Users can also comment on comments itself. This will create an indentation on the GUI to suggest that the parent comment has a child comment. When a comment is commented upon, the attribute *comment_comment_id* will insert the parent *comment_id* which shows the relation of two comments, one parent and the other being the child.

7.1.9 user, is_in_message, private_message table

Another functionality found in Turtlenet is the user is able to send private messages to users. When a private message is created by the main user, a new tuple is added into the *private_message* table. The user then has the option to add other user(s) into the conversation. When done so, a tuple or tuples, depending on the number of users he has added onto the conversation, are added into the *is_in_message* table. This inserts the information such as the time of when the user has been added into the conversation, the user's ID and message ID. The *private_message* table on the other hand stores data such as the content of the message and the time for which this whole conversation was created.

7.1.10 message_claim table

This table stores all CLAIM messages which cannot be matched with a public key. When a new key is entered we search for the CLAIM message, erase it, and add a new entry to the user table.

7.1.11 key_revoke table

This stores key revocation messages. If a user suspects that their private key has been compromised then they can send a message informing their relations of this. Once a key revocation message is sent all content posted after the given time and signed with the corresponding private key is marked as untrusted.

7.1.12 login_logout_log table

This table simply tracks the login and logout activities of the main user. When a user logs in and out, a new tuple will be inserted into this table.

7.2 Table layout of the database

NB: Public keys are 217 characters long, all id's are auto-incremented.

Table 7.1: user

Name	Datatype	Key
user_id	INT	PK
username	VARCHAR(25)	
name	VARCHAR(30)	
birthday	DATE	
sex	VARCHAR(1)	
email	VARCHAR(30)	
public_key	VARCHAR(600)	PK

Table 7.2: is_in_category

Name	Datatype	Key
is_in_id	INT	PK
category_id	INT	FK
user_id	INT	FK

Table 7.3: category

Name	Datatype	Key
category_id	INT	PK
name	VARCHAR(30)	

Table 7.4: private_message

Name	Datatype	Key
message_id	INT	PK
from	INT	
content	VARCHAR(50)	
time	DATE	

Table 7.5: is_in_message

Name	Datatype	Key
is_in_id	INT	PK
time	DATETIME	
message_id	INT	FK
user_id	INT	FK

Table 7.6: wall_post

Name	Datatype	Key
wall_id	INT	PK
from	INT	FK
to	INT	FK
content	VARCHAR(50)	
time	DATETIME	

Table 7.7: allowed_to

Name	Datatype	Key
allowed_to_id	INT	PK
user_id	INT	FK
category_id	INT	FK
post_id	INT	FK

Table 7.8: has_comment

Name	Datatype	Key
comment_id	INT(100)	PK
comment_content	VARCHAR(50)	
post_id	INT(100)	FK
user_id	VARCHAR(50)	FK
comment_comment_id	INT(100)	FK
time	DATETIME	

Table 7.9: has_like

Name	Datatype	Key
like_id	INT	PK
post_id	INT	FK
user_id	INT	FK
comment_id	INT	FK
time	DATETIME	

Table 7.10: events

Name	Datatype	Key
event_id	INT	PK
title	VARCHAR(10)	
content	VARCHAR(40)	
time	DATETIME	
start_date	DATETIME	
end_date	DATETIME	
from	INT	FK

Table 7.11: is_invited

Name	Datatype	Key
is_invited_id	INT	PK
user_id	INT	FK
is_in_category_id	INT	FK
event_id	INT	FK
decision	BIT	

Table 7.12: login_logout_log

Name	Datatype	Key
log_id	INT	PK
login_time	DATETIME	
logout_time	DATETIME	

Table 7.13: key_revoke

Name	Datatype	Key
revoke_id	INT	PK
signature	VARCHAR(45)	
time	DATETIME	

Table 7.14: message_claim

Name	Datatype	Key
username	VARCHAR(25)	PK
signature	VARCHAR(45)	

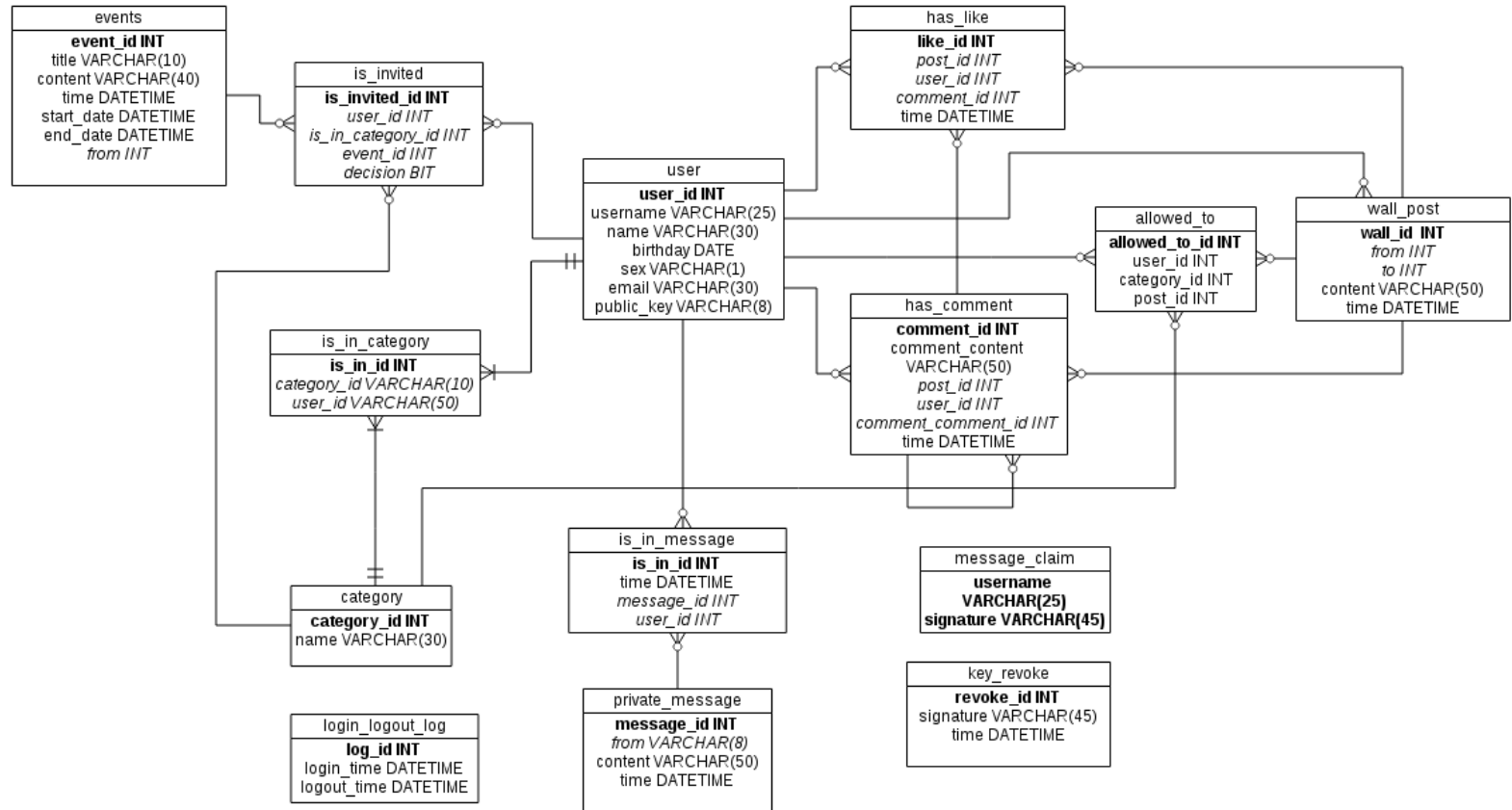


Figure 7.1: Database Entity Relationship diagram

Chapter 8

Transaction details

The table below shows the transaction details of each function which will be found in the program. There are four types of transactions for databases which are insert, read, update and delete.

Insertion is done when new data is added into a NULL attribute. Read on the other hand, is to view information from selected table(s) and its attribute(s). Similar as insertion but update is conducted when data already exists in the particular attribute. This basically removes previous data and add a new one. Lastly, delete, as it is self explanatory, deletes the whole tuple from the database. However this is usually avoided in database norms.

Function	Table(s) involved	Transaction(s)
addClaim()	message_claim	Insert
getClaims	message_claim	Read
getUsernames	user	Read
addRevocation	key_revoke	Insert
getRevocations	key_revoke	Read
isRevoked()	key_revoke	Read
addPData()	user	Update
getPData()	user	Read
createChat()	private_message	Insert
getChat()	private_message, is_in_message	Read
addToChat()	is_in_message	Insert
addPost()	wall_post	Insert
getPosts()	wall_post	Read

Function	Table(s) involved	Transaction(s)
addComment()	has_comment, wall_post	Insert, Read
getComments()	has_comment, wall_post	Read
addLike()	has_like, wall_post, has_comment	Insert, Read
getLikes()	has_like, wall_post, has_comment, user	Read
countLikes()	has_like, wall_post, has_comment	Read
addEvent()	events	Insert
getEvent()	events	Read
acceptEvent()	events	Update
declineEvent()	events	Update
addKey()	message_claim, user	Read, Delete, Insert
getKey()	user	Read
getName()	user	Read
addCategory()	category	Insert
addToCategory()	category, is_in_category	Read, Insert

Chapter 9

User Interfaces

9.1 Interface Research

As a social network, the user interface design is of high importance, as a lot of users of the program will have little core system knowledge, and rely entirely on the user interface. As a result we have looked at a variety of options into designing which will be the best for the project.

9.1.1 Swing

Swing is the primary Java GUI toolkit, providing a basic standpoint for entry level interface designing. Introduced back in 1996, Swing was designed to be an interface style that required minimal changes to the applications code, providing the user with a pluggable look and feel mechanism. It has been apart of the standard java library for over a decade, which, as I will now explain, may not be to our benefit.

Swing, whilst an excellent language to begin with, and write simple applications in, is quite dated. As our group advisor put it when inquiring about what we would be coding the user interface in:

"You should avoid Swing to prevent it looking like it was done in the nineties." - Sebastian Coope

Sebastian is not wrong either, as Swing does a very plain feel to it. This figure shows an old instant messaging system written with Swing by one of our team members. As you can see it is unlikely to appeal to the mass market with such visually plain appearance. This makes Swing, unlikely to be our GUI toolkit of choice, despite some of our members experience with it.

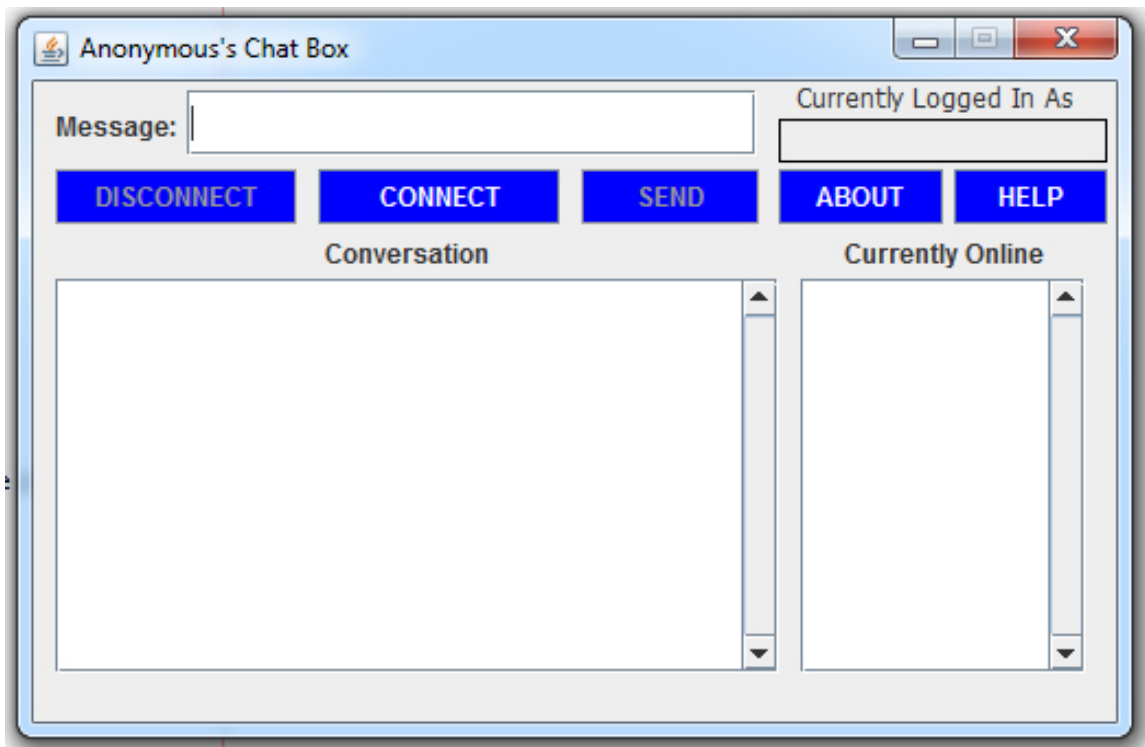


Figure 9.1: Swing Instant Messaging Application

9.1.2 Abstract Window Toolkit

Abstract Window Toolkit (otherwise known as AWT), was another choice given that we are programming in Java, and synchronicity between the two would be an advantage. Whilst AWT retained some advantages such as its style blending in with each operating system it runs on, it is even older than Swing being Java's original toolkit, making any GUI displayed via it look rather dated. None of the the current team has any proficiency with AWT however, and whilst it is possible to learn, there are still other options to consider that may provide the use with a more professional GUI build.

9.1.3 Standard Widget Toolkit

Standard Widget Toolkit (otherwise known as SWT), is one of the more promising candidates so far given its look and up-to-date support packages. The latest stable release of SWT was only last year, and is capable of producing programs with a modern and professionally built appearance, as shown in the figure.

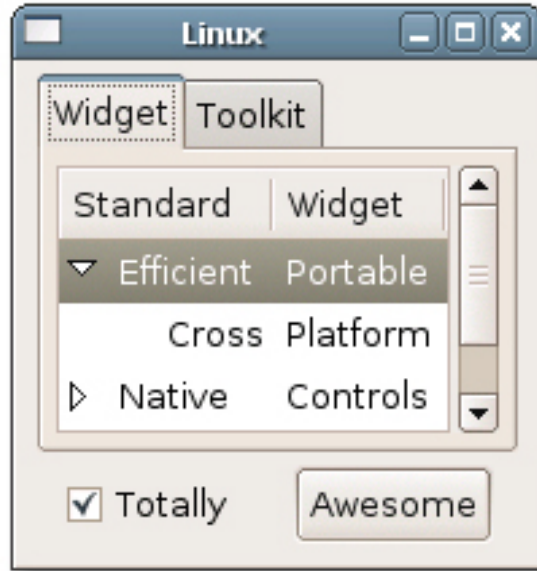


Figure 9.2: SWT Appearance Style

Unlike both Swing and AWT, SWT is not provided by Sun Microsystems as a part of the Java platform. It is now provided and maintained by the Eclipse Foundation, and provided as a part of their widely used Eclipse IDE, something a lot of the team is familiar with.

9.1.4 GWT

GWT allows you to create HTML/Javascript based user interfaces for Java applications running locally. The interface is programmed in Java and then GWT creates valid HTML/Javascript automatically. A web server is required in order for Javascript events to be sent to the Java application.

The user can then interact with the system by pointing their web browser at localhost. This has the benefit of being familiar to novice users as most modern computer interaction is done within a web browser.

Another advantage of using GWT is the ability to alter the appearance of web pages using CSS. This facilitates the creation of a modern, attractive user interface that integrates nicely with current operating systems and software.

9.1.5 Javascript

It is possible to create the entire client application in Javascript and use a HTML/Javascript GUI. This approach removes the need for a local web server meaning the only software the user is required

to run is a modern web browser.

Another advantage would be tight integration between the logic and interface elements of the client application and no risk of errors caused by using multiple programming languages.

One disadvantage of this approach is the difficulty in implementing the required security measures and encryption in Javascript. This can be remedied by using a Javascript library such as the Forge project which implements many cryptography methods.

The main disadvantage is that in this approach the server operator has complete control of the client the user uses. This is unacceptable because we're assuming that the server operator is seeking to spy on the user.

9.2 GUI Design

9.2.1 Client Design

Arguably the most important GUI in the project is the client GUI, as this is what the standard user will be interacting with, a person whom we are assuming has no knowledge of any inner workings. All tests we perform on our system at a later stage will be through this client, as per such its design takes a high level of importance. Its for this reason we have chosen something common users will be more accustomed to: web pages.

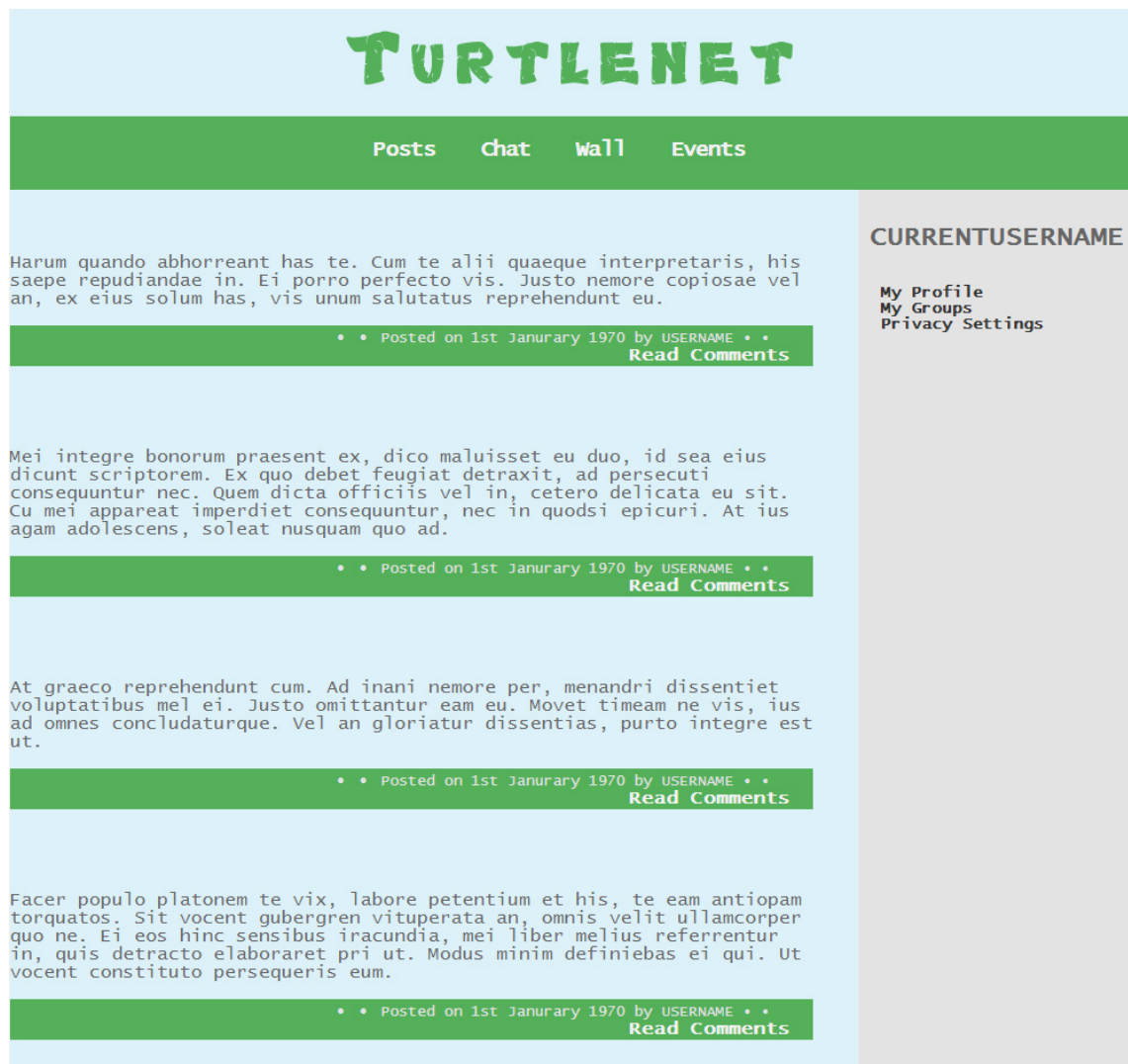


Figure 9.3: Client Design Image

Most users will be familiar with HTML and CSS page layouts, even if they do not know what HTML or CSS is. This will provide a certain level of comfort when it comes to using new applications and how to navigate between pages or tabs. Javascript would be used to pipe the data to the client program, but this is something the user would not interact with or see. It also provides the advantage of knowing nearly every operating system nowadays comes with a web browser natively meaning a HTML/CSS based GUI would likely be supported on nearly all platforms. For these reasons we have chosen to use GWT. A local web server has been decided as the best way forward, as it will

provide the best form of security from the server operators.

9.2.2 Server Design

Whilst not critically important, as it would only be operated by those with technical knowledge, is still an important aspect to consider. It needs to hold the system level settings and control mechanisms a server client would need, whilst not making them immediately and 'accidentally' accessible via the form of large obvious buttons. The easiest way of doing this is via a command input box beneath a chat log window to provide commands that way. It is also may be an idea to show server data such as memory usage on the operators end, as this data is completely accessible and non-intrusive to the client. The figure labelled 'Server Design Image' shows an example of how the server client may be completed. Pending on the features allowed in GWT, our method of choice, we will aim for it to retain a similar appearance.

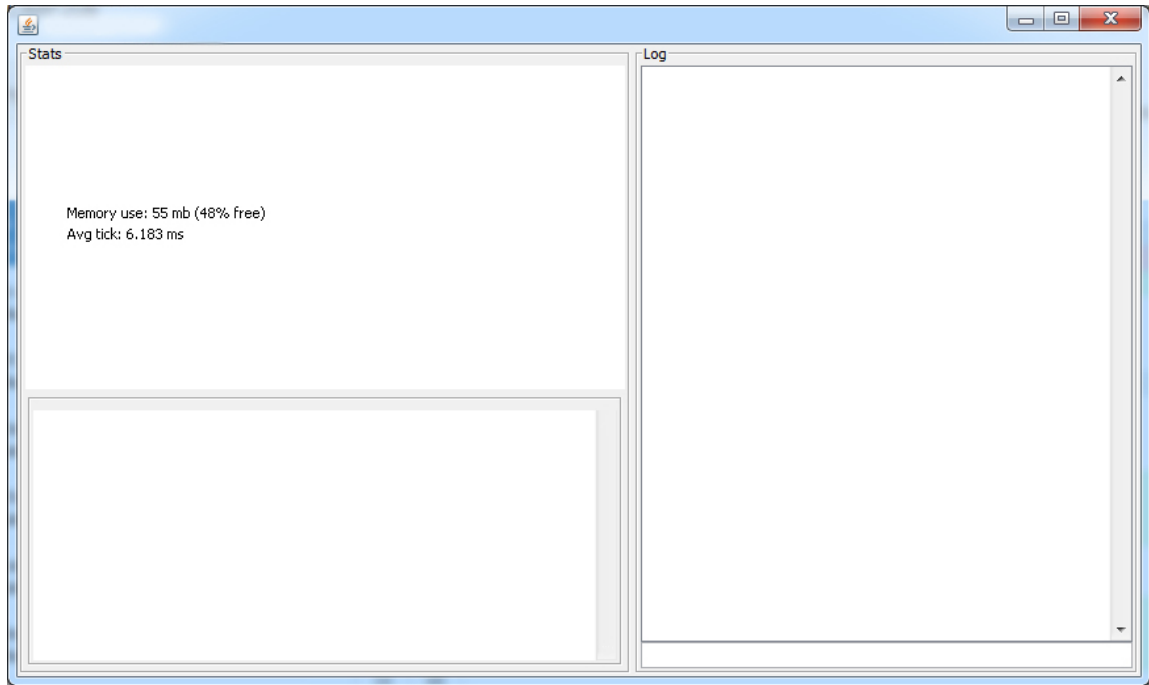


Figure 9.4: Server Design Image

9.3 Future Work

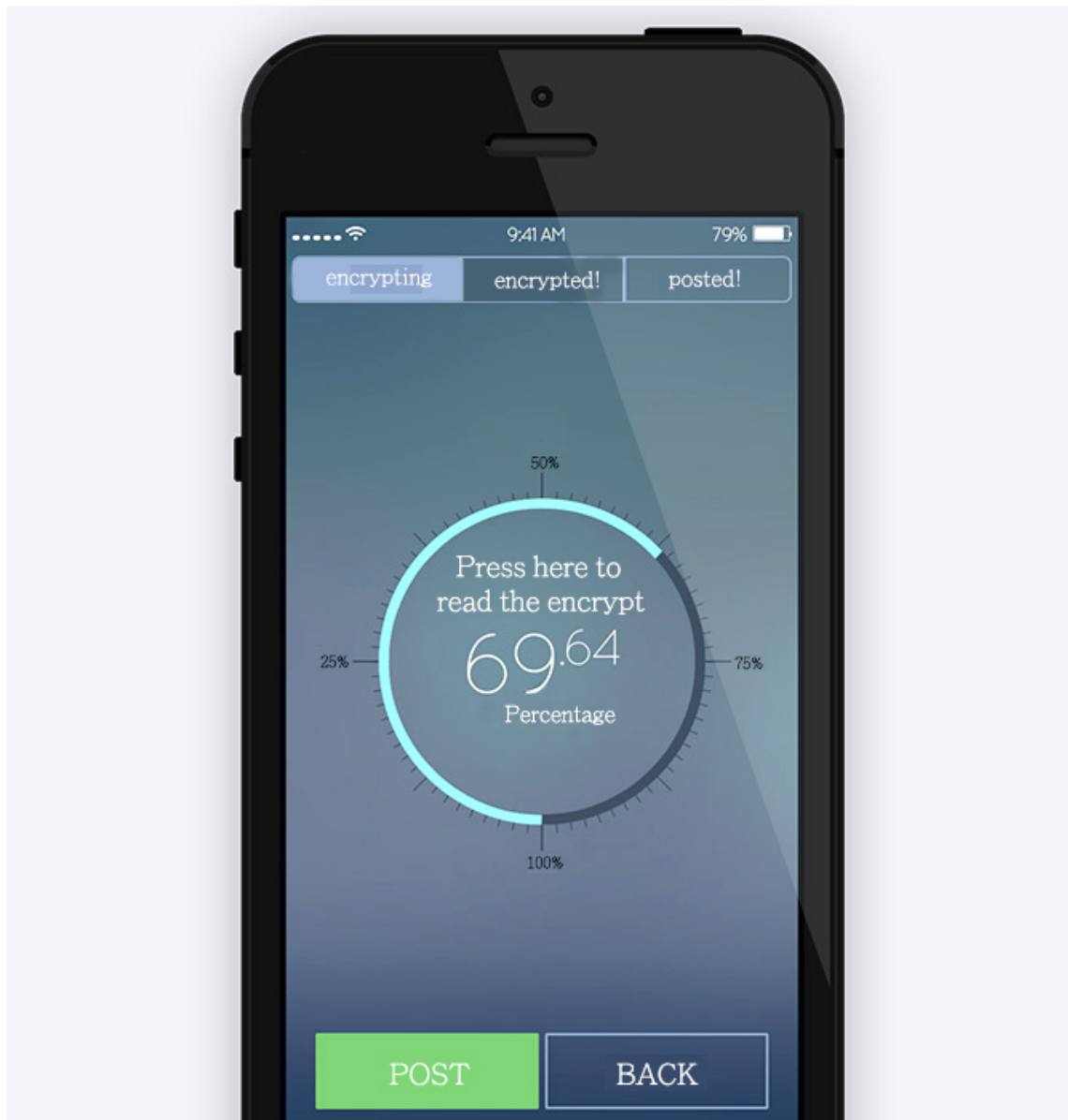


Figure 9.5: Mobile Sending Stage GUI

With some of the spare resources available during this phase, we were able to look into some future design work on the mobile front. One of our designers had some experience in this field of work

and offered to put some images together of what a mobile application version of our product could potentially look like.

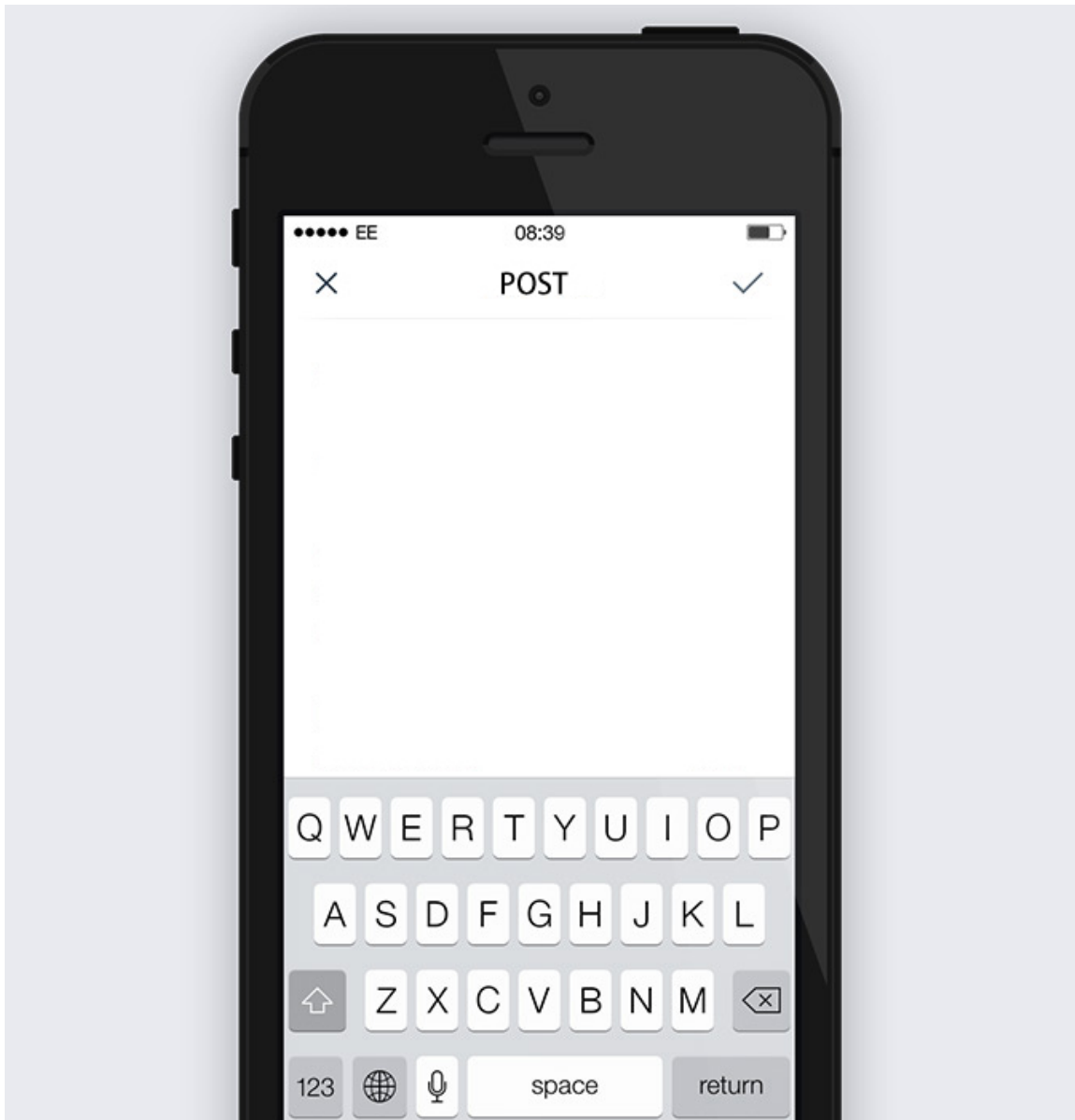


Figure 9.6: Mobile Post Stage GUI

The mobile interface data flow diagram shows how the application would flow between screens, giving an idea to the level of depth an application of this size might have.

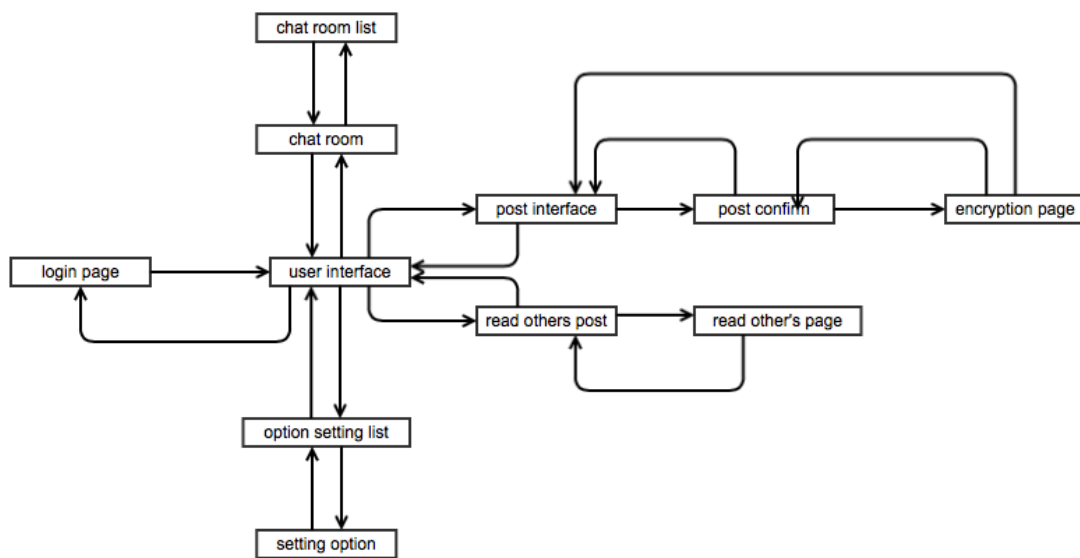


Figure 9.7: Mobile Data Flow Diagram

Chapter 10

Business Rules

In standard projects the business model can commonly outline certain validation practices for the program or project in the form of business rules or policies. Our project, and its fundamental idea, works a little different than most projects in terms of business, as per such we have only one business rule.

- To ensure the client never sends identifying data to the server or its operators.

This is to ensure that the privacy of communication is always within the hands of the client and user, as opposed to any who run the network. To violate this single rule would be going against both the company ideals, and the projects goals.

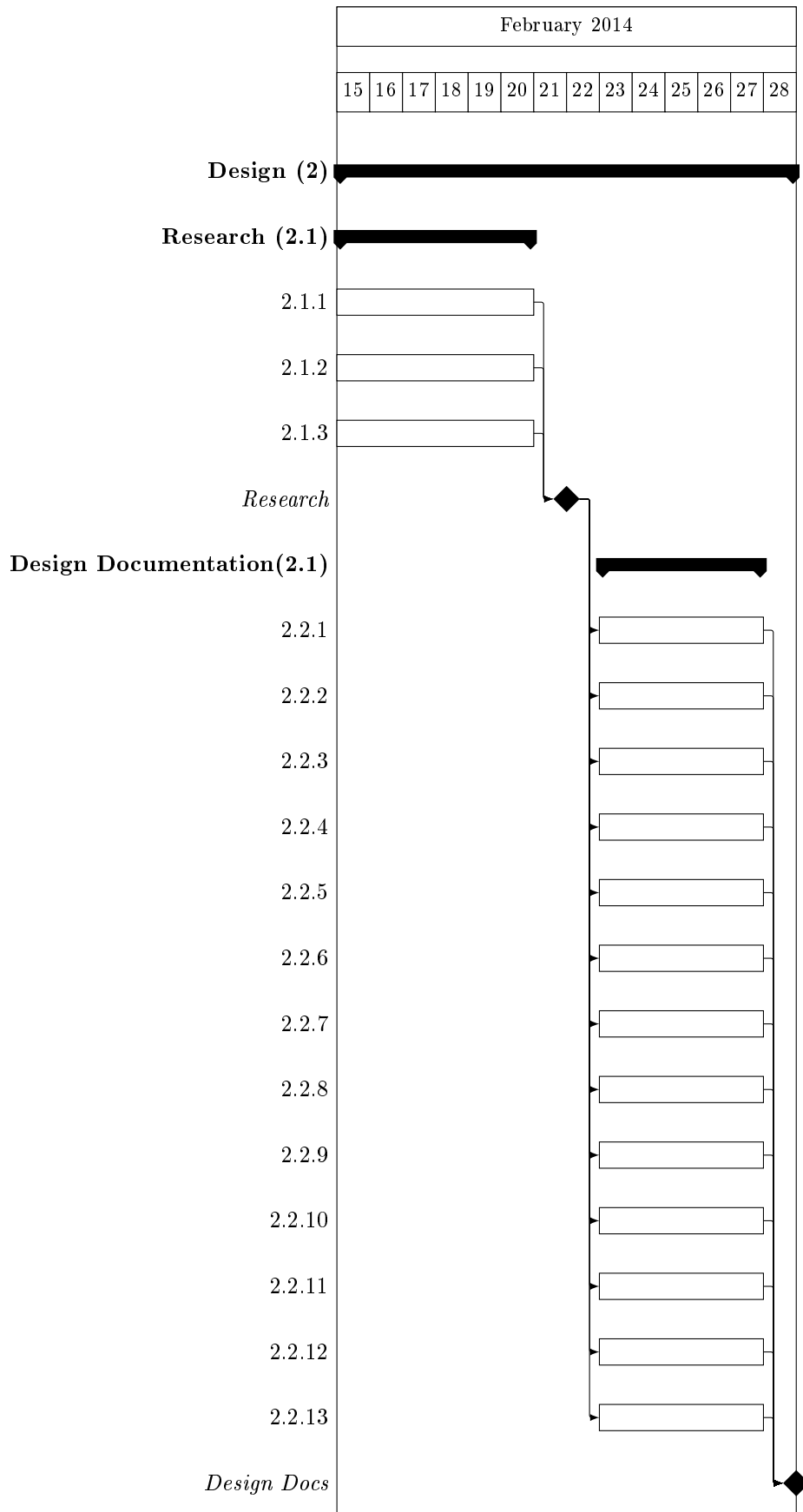
Chapter 11

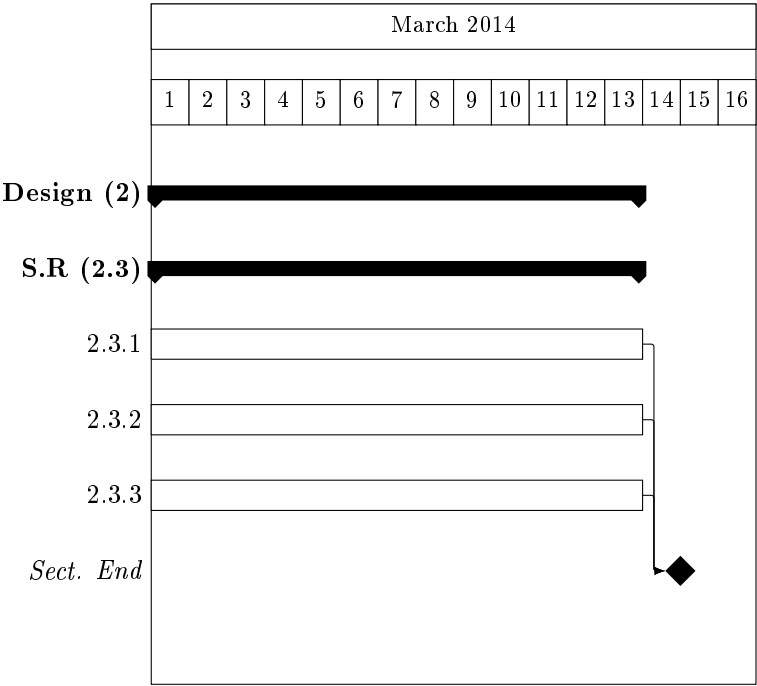
Gantt Chart

These are excerpts from the Gantt charts made during the requirements stage of the project. They have been used as a guide throughout the completed sections of the project. As the design section was foreseen as the most work-intensive part that is encompassed within the project, particular care and attention was made to make sure that official deadlines were met, through the use of un-official end dates for each task.

By doing so, therefore finishing tasks earlier than required, it has provided a buffer used for quality controlling the project's deliverables. A partial reproduction of the task list is also provided:

Task ID	Task Description (Desc.)	Due Date	Deliverable
2	Project Design	14/03/2014	Design Segment
2.1	Research (Res.)	21/02/2014	Research Segment
2.1.1	Res: Database Languages	21/02/2014	Same as Desc.
2.1.2	Res: Programming Languages	21/02/2014	Same as Desc.
2.1.3	Res: Interfaces	21/02/2014	Same as Desc.
2.2	Designs (Des.)	07/03/2014	Design Segment
2.2.1	Des: Databases	28/02/2014	Same as Desc.
2.2.2	Des: Class Interfaces	28/02/2014	Same as Desc.
2.2.3	Des: Protocol	28/02/2014	Same as Desc.
2.2.4	Des: Architecture	28/02/2014	Same as Desc.
2.2.5	Des: Sequence Diagrams	28/02/2014	Same as Desc.
2.2.6	Des: Data Flow Diagrams	28/02/2014	Same as Desc.
2.2.7	Des: Class Diagrams	28/02/2014	Same as Desc.
2.2.8	Des: Server-side Interfaces	28/02/2014	Same as Desc.
2.2.9	Des: Client-side Interfaces	28/02/2014	Same as Desc.
2.2.10	Des: Server-side Protocols	28/02/2014	Same as Desc.
2.2.11	Des: Client-side Protocols	28/02/2014	Same as Desc.
2.2.12	Des: Server-side Pseudo-code	07/03/2014	Same as Desc.
2.2.13	Des: Client-side Pseudo-code	07/03/2014	Same as Desc.
2.3	Segment Review	10/03/2014	Design Segment
2.3.1	Evaluate Segment Quality	14/03/2014	N/A
2.3.2	Improve Segment	14/03/2014	Design Segment





Chapter 12

Glossary

AES - Symmetric encryption standard.

Category - We allow our users to create ‘Categories’, and place one or more users into one or more categories. These sets of users are used to speed up repetitive actions such as allowing all of your friends permission to view something, by instead allowing the user to allow the category ‘friends’ to view it.

Client - The program that will be used by users which connects to a turtlesnet server.

Facebook - A social networking website designed to make the world more open and to connect people together in a simple format.

Onion Routing - A manner of routing traffic in a network with the goal of obscuring from the recipient who the sender was. This is achieved by routing it through a number of intermediaries, none of which have access to both who sent the traffic, and the plaintext traffic. ¹

Privacy - Personal information being known to only those whom you choose to inform of it.

Parameterized Query - A precompiled query lacking important information for the values of parts of it. These are used to protect against SQL injection and to provide a greater degree of abstraction from the database for the rest of the system.

QR Code - QR stands for Quick Response. Used to store data it is a form of 2D bar code, It was designed to be easy to read from low quality photographs.

Relation - Two users must know each others public keys in order to communicate. We say that two users who possess each others keys are in a ‘relation’. This is done because it is a situation we talk about often, and it helps to have a word for it.

RSA - An asymmetric encryption algorithm.

SocialNetwork - A website build around facillitating social interaction.

Server - A computer running the turtlesnet server which allows clients to connect to it.

¹See http://en.wikipedia.org/wiki/Onion_routing for more information.

ServerOperator - The owners and engineers responsible for running Turtlenet servers.

Tor - An implementation of onion routing.

Appendices

Appendix A

Deadlines

- **2014-01-31** topic and team
- **2014-02-14** requirements
- **2014-03-14** design
- **2014-05-09** portfolio & individual submission

Appendix B

Licence



To the extent possible under law, Ballmer Peak has waived all copyright and related or neighboring rights to Turtlenet and Associated Documentation. This work is published from:
United Kingdom.

B.1 Statement of Purpose

The laws of most jurisdictions throughout the world automatically confer exclusive Copyright and Related Rights (defined below) upon the creator and subsequent owner(s) (each and all, an "owner") of an original work of authorship and/or a database (each, a "Work").

Certain owners wish to permanently relinquish those rights to a Work for the purpose of contributing to a commons of creative, cultural and scientific works ("Commons") that the public can reliably and without fear of later claims of infringement build upon, modify, incorporate in other works, reuse and redistribute as freely as possible in any form whatsoever and for any purposes, including without limitation commercial purposes. These owners may contribute to the Commons to promote the ideal of a free culture and the further production of creative, cultural and scientific works, or to gain reputation or greater distribution for their Work in part through the use and efforts of others.

For these and/or other purposes and motivations, and without any expectation of additional consideration or compensation, the person associating CC0 with a Work (the "Affirmer"), to the extent that he or she is an owner of Copyright and Related Rights in the Work, voluntarily elects

to apply CC0 to the Work and publicly distribute the Work under its terms, with knowledge of his or her Copyright and Related Rights in the Work and the meaning and intended legal effect of CC0 on those rights.

B.2 Copyright and Related Rights

A Work made available under CC0 may be protected by copyright and related or neighboring rights ("Copyright and Related Rights"). Copyright and Related Rights include, but are not limited to, the following:

1. the right to reproduce, adapt, distribute, perform, display, communicate, and translate a Work;
2. moral rights retained by the original author(s) and/or performer(s);
3. publicity and privacy rights pertaining to a person's image or likeness depicted in a Work;
4. rights protecting against unfair competition in regards to a Work, subject to the limitations in paragraph 4(a), below;
5. rights protecting the extraction, dissemination, use and reuse of data in a Work;
6. database rights (such as those arising under Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, and under any national implementation thereof, including any amended or successor version of such directive); and
7. other similar, equivalent or corresponding rights throughout the world based on applicable law or treaty, and any national implementations thereof.

B.3 Waiver

To the greatest extent permitted by, but not in contravention of, applicable law, Affirmer hereby overtly, fully, permanently, irrevocably and unconditionally waives, abandons, and surrenders all of Affirmer's Copyright and Related Rights and associated claims and causes of action, whether now known or unknown (including existing as well as future claims and causes of action), in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "Waiver"). Affirmer makes the Waiver for the benefit of each member

of the public at large and to the detriment of Affirmer's heirs and successors, fully intending that such Waiver shall not be subject to revocation, rescission, cancellation, termination, or any other legal or equitable action to disrupt the quiet enjoyment of the Work by the public as contemplated by Affirmer's express Statement of Purpose.

B.4 Public License Fallback

Should any part of the Waiver for any reason be judged legally invalid or ineffective under applicable law, then the Waiver shall be preserved to the maximum extent permitted taking into account Affirmer's express Statement of Purpose. In addition, to the extent the Waiver is so judged Affirmer hereby grants to each affected person a royalty-free, non transferable, non sublicensable, non exclusive, irrevocable and unconditional license to exercise Affirmer's Copyright and Related Rights in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "License"). The License shall be deemed effective as of the date CC0 was applied by Affirmer to the Work. Should any part of the License for any reason be judged legally invalid or ineffective under applicable law, such partial invalidity or ineffectiveness shall not invalidate the remainder of the License, and in such case Affirmer hereby affirms that he or she will not (i) exercise any of his or her remaining Copyright and Related Rights in the Work or (ii) assert any associated claims and causes of action with respect to the Work, in either case contrary to Affirmer's express Statement of Purpose.

B.5 Limitations and Disclaimers

1. No trademark or patent rights held by Affirmer are waived, abandoned, surrendered, licensed or otherwise affected by this document.
2. Affirmer offers the Work as-is and makes no representations or warranties of any kind concerning the Work, express, implied, statutory or otherwise, including without limitation warranties of title, merchantability, fitness for a particular purpose, non infringement, or the absence of latent or other defects, accuracy, or the present or absence of errors, whether or not discoverable, all to the greatest extent permissible under applicable law.
3. Affirmer disclaims responsibility for clearing rights of other persons that may apply to the Work or any use thereof, including without limitation any person's Copyright and Related Rights in the Work. Further, Affirmer disclaims responsibility for obtaining any necessary consents, permissions or other rights required for any use of the Work.

4. Affirmer understands and acknowledges that Creative Commons is not a party to this document and has no duty or obligation with respect to this CC0 or use of the Work.

B.6 Included Works

We did not write or create the following:

- writeup/latex/tikz-uml.sty
- writeup/latex/todonotes.sty
- writeup/latex/ulem.sty
- tikz-styles.sty
- writeup/images/appendicies/licence.png (CC0 licence logo)
- The text of any legal licence
- client/web_interface_mockup/jquery.js
- client/web_interface_mockup/turtles.ttf

B.7 jquery.js Licence

Copyright 2014 jQuery Foundation and other contributors <http://jquery.com/>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

B.8 `todonotes.sty` licence

Copyright (c) 2008 by Henrik Skov Midtby <henrikmidtby@gmail.com>

This file may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version 1.2 of this license or (at your option) any later version. This licence is provided below.

B.9 The LaTeX Project Public License

LPPL Version 1.3c 2008-05-04

Copyright 1999 2002-2008 LaTeX3 Project

Everyone is allowed to distribute verbatim copies of this
license document, but modification of it is not allowed.

B.9.1 PREAMBLE

The LaTeX Project Public License (LPPL) is the primary license under which the LaTeX kernel and the base LaTeX packages are distributed.

You may use this license for any work of which you hold the copyright and which you wish to distribute. This license may be particularly suitable if your work is TeX-related (such as a LaTeX package), but it is written in such a way that you can use it even if your work is unrelated to TeX.

The section ‘WHETHER AND HOW TO DISTRIBUTE WORKS UNDER THIS LICENSE’, below, gives instructions, examples, and recommendations for authors who are considering distributing their works under this license.

This license gives conditions under which a work may be distributed and modified, as well as conditions under which modified versions of that work may be distributed.

We, the LaTeX3 Project, believe that the conditions below give you the freedom to make and distribute modified versions of your work that conform with whatever technical specifications you wish while maintaining the availability, integrity, and reliability of that work. If you do not see how to achieve your goal while meeting these conditions, then read the document ‘`cfgguide.tex`’ and ‘`modguide.tex`’ in the base LaTeX distribution for suggestions.

B.9.2 DEFINITIONS

In this license document the following terms are used:

‘**Work**’ Any work being distributed under this License.

‘**Derived Work**’ Any work that under any applicable law is derived from the Work.

‘Modification’ Any procedure that produces a Derived Work under any applicable law – for example, the production of a file containing an original file associated with the Work or a significant portion of such a file, either verbatim or with modifications and/or translated into another language.

‘Modify’ To apply any procedure that produces a Derived Work under any applicable law.

‘Distribution’ Making copies of the Work available from one person to another, in whole or in part. Distribution includes (but is not limited to) making any electronic components of the Work accessible by file transfer protocols such as FTP or HTTP or by shared file systems such as Sun’s Network File System (NFS).

‘Compiled Work’ A version of the Work that has been processed into a form where it is directly usable on a computer system. This processing may include using installation facilities provided by the Work, transformations of the Work, copying of components of the Work, or other activities. Note that modification of any installation facilities provided by the Work constitutes modification of the Work.

‘Current Maintainer’ A person or persons nominated as such within the Work. If there is no such explicit nomination then it is the ‘Copyright Holder’ under any applicable law.

‘Base Interpreter’ A program or process that is normally needed for running or interpreting a part or the whole of the Work.

A Base Interpreter may depend on external components but these are not considered part of the Base Interpreter provided that each external component clearly identifies itself whenever it is used interactively. Unless explicitly specified when applying the license to the Work, the only applicable Base Interpreter is a ‘LaTeX-Format’ or in the case of files belonging to the ‘LaTeX-format’ a program implementing the ‘TeX language’.

B.9.3 CONDITIONS ON DISTRIBUTION AND MODIFICATION

1. Activities other than distribution and/or modification of the Work are not covered by this license; they are outside its scope. In particular, the act of running the Work is not restricted and no requirements are made concerning any offers of support for the Work.
2. You may distribute a complete, unmodified copy of the Work as you received it. Distribution of only part of the Work is considered modification of the Work, and no right to distribute such a Derived Work may be assumed under the terms of this clause.
3. You may distribute a Compiled Work that has been generated from a complete, unmodified copy of the Work as distributed under Clause 2 above, as long as that Compiled Work is

distributed in such a way that the recipients may install the Compiled Work on their system exactly as it would have been installed if they generated a Compiled Work directly from the Work.

4. If you are the Current Maintainer of the Work, you may, without restriction, modify the Work, thus creating a Derived Work. You may also distribute the Derived Work without restriction, including Compiled Works generated from the Derived Work. Derived Works distributed in this manner by the Current Maintainer are considered to be updated versions of the Work.
5. If you are not the Current Maintainer of the Work, you may modify your copy of the Work, thus creating a Derived Work based on the Work, and compile this Derived Work, thus creating a Compiled Work based on the Derived Work.
6. If you are not the Current Maintainer of the Work, you may distribute a Derived Work provided the following conditions are met for every component of the Work unless that component clearly states in the copyright notice that it is exempt from that condition. Only the Current Maintainer is allowed to add such statements of exemption to a component of the Work.
 - a. If a component of this Derived Work can be a direct replacement for a component of the Work when that component is used with the Base Interpreter, then, wherever this component of the Work identifies itself to the user when used interactively with that Base Interpreter, the replacement component of this Derived Work clearly and unambiguously identifies itself as a modified version of this component to the user when used interactively with that Base Interpreter.
 - b. Every component of the Derived Work contains prominent notices detailing the nature of the changes to that component, or a prominent reference to another file that is distributed as part of the Derived Work and that contains a complete and accurate log of the changes.
 - c. No information in the Derived Work implies that any persons, including (but not limited to) the authors of the original version of the Work, provide any support, including (but not limited to) the reporting and handling of errors, to recipients of the Derived Work unless those persons have stated explicitly that they do provide such support for the Derived Work.
 - d. You distribute at least one of the following with the Derived Work:
 - (a) A complete, unmodified copy of the Work; if your distribution of a modified component is made by offering access to copy the modified component from a designated place, then offering equivalent access to copy the Work from the same or some similar place meets this condition, even though third parties are not compelled to copy the Work along with the modified component;

- (b) Information that is sufficient to obtain a complete, unmodified copy of the Work.
- 7. If you are not the Current Maintainer of the Work, you may distribute a Compiled Work generated from a Derived Work, as long as the Derived Work is distributed to all recipients of the Compiled Work, and as long as the conditions of Clause 6, above, are met with regard to the Derived Work.
- 8. The conditions above are not intended to prohibit, and hence do not apply to, the modification, by any method, of any component so that it becomes identical to an updated version of that component of the Work as it is distributed by the Current Maintainer under Clause 4, above.
- 9. Distribution of the Work or any Derived Work in an alternative format, where the Work or that Derived Work (in whole or in part) is then produced by applying some process to that format, does not relax or nullify any sections of this license as they pertain to the results of applying that process.
- 10. **a.** A Derived Work may be distributed under a different license provided that license itself honors the conditions listed in Clause 6 above, in regard to the Work, though it does not have to honor the rest of the conditions in this license.
- b.** If a Derived Work is distributed under a different license, that Derived Work must provide sufficient documentation as part of itself to allow each recipient of that Derived Work to honor the restrictions in Clause 6 above, concerning changes from the Work.
- 11. This license places no restrictions on works that are unrelated to the Work, nor does this license place any restrictions on aggregating such works with the Work by any means.
- 12. Nothing in this license is intended to, or may be used to, prevent complete compliance by all parties with all applicable laws.

B.9.4 NO WARRANTY

There is no warranty for the Work. Except when otherwise stated in writing, the Copyright Holder provides the Work ‘as is’, without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the Work is with you. Should the Work prove defective, you assume the cost of all necessary servicing, repair, or correction.

In no event unless required by applicable law or agreed to in writing will The Copyright Holder, or any author named in the components of the Work, or any other party who may distribute and/or modify the Work as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of any use of the Work or out of inability to use

the Work (including, but not limited to, loss of data, data being rendered inaccurate, or losses sustained by anyone as a result of any failure of the Work to operate with any other programs), even if the Copyright Holder or said author or said other party has been advised of the possibility of such damages.

B.9.5 MAINTENANCE OF THE WORK

The Work has the status ‘author-maintained’ if the Copyright Holder explicitly and prominently states near the primary copyright notice in the Work that the Work can only be maintained by the Copyright Holder or simply that it is ‘author-maintained’.

The Work has the status ‘maintained’ if there is a Current Maintainer who has indicated in the Work that they are willing to receive error reports for the Work (for example, by supplying a valid e-mail address). It is not required for the Current Maintainer to acknowledge or act upon these error reports.

The Work changes from status ‘maintained’ to ‘unmaintained’ if there is no Current Maintainer, or the person stated to be Current Maintainer of the work cannot be reached through the indicated means of communication for a period of six months, and there are no other significant signs of active maintenance.

You can become the Current Maintainer of the Work by agreement with any existing Current Maintainer to take over this role.

If the Work is unmaintained, you can become the Current Maintainer of the Work through the following steps:

1. Make a reasonable attempt to trace the Current Maintainer (and the Copyright Holder, if the two differ) through the means of an Internet or similar search.
2. If this search is successful, then enquire whether the Work is still maintained.
 - a. If it is being maintained, then ask the Current Maintainer to update their communication data within one month.
 - b. If the search is unsuccessful or no action to resume active maintenance is taken by the Current Maintainer, then announce within the pertinent community your intention to take over maintenance. (If the Work is a LaTeX work, this could be done, for example, by posting to comp.text.tex.)
3.
 - a. If the Current Maintainer is reachable and agrees to pass maintenance of the Work to you, then this takes effect immediately upon announcement.
 - b. If the Current Maintainer is not reachable and the Copyright Holder agrees that maintenance of the Work be passed to you, then this takes effect immediately upon announcement.

4. If you make an ‘intention announcement’ as described in 2b. above and after three months your intention is challenged neither by the Current Maintainer nor by the Copyright Holder nor by other people, then you may arrange for the Work to be changed so as to name you as the (new) Current Maintainer.
5. If the previously unreachable Current Maintainer becomes reachable once more within three months of a change completed under the terms of 3b) or 4), then that Current Maintainer must become or remain the Current Maintainer upon request provided they then update their communication data within one month.

A change in the Current Maintainer does not, of itself, alter the fact that the Work is distributed under the LPPL license.

If you become the Current Maintainer of the Work, you should immediately provide, within the Work, a prominent and unambiguous statement of your status as Current Maintainer. You should also announce your new status to the same pertinent community as in 2b) above.

B.9.6 WHETHER AND HOW TO DISTRIBUTE WORKS UNDER THIS LICENSE

This section contains important instructions, examples, and recommendations for authors who are considering distributing their works under this license. These authors are addressed as ‘you’ in this section.

Choosing This License or Another License

If for any part of your work you want or need to use *distribution* conditions that differ significantly from those in this license, then do not refer to this license anywhere in your work but, instead, distribute your work under a different license. You may use the text of this license as a model for your own license, but your license should not refer to the LPPL or otherwise give the impression that your work is distributed under the LPPL.

The document ‘modguide.tex’ in the base LaTeX distribution explains the motivation behind the conditions of this license. It explains, for example, why distributing LaTeX under the GNU General Public License (GPL) was considered inappropriate. Even if your work is unrelated to LaTeX, the discussion in ‘modguide.tex’ may still be relevant, and authors intending to distribute their works under any license are encouraged to read it.

A Recommendation on Modification Without Distribution

It is wise never to modify a component of the Work, even for your own personal use, without also meeting the above conditions for distributing the modified component. While you might intend

that such modifications will never be distributed, often this will happen by accident – you may forget that you have modified that component; or it may not occur to you when allowing others to access the modified version that you are thus distributing it and violating the conditions of this license in ways that could have legal implications and, worse, cause problems for the community. It is therefore usually in your best interest to keep your copy of the Work identical with the public one. Many works provide ways to control the behavior of that work without altering any of its licensed components.

How to Use This License

To use this license, place in each of the components of your work both an explicit copyright notice including your name and the year the work was authored and/or last substantially modified. Include also a statement that the distribution and/or modification of that component is constrained by the conditions in this license.

Here is an example of such a notice and statement:

```
%% pig.dtx
%% Copyright 2005 M. Y. Name
%
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in
%   http://www.latex-project.org/lppl.txt
% and version 1.3 or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
%
% This work has the LPPL maintenance status ‘maintained’.
%
% The Current Maintainer of this work is M. Y. Name.
%
% This work consists of the files pig.dtx and pig.ins
% and the derived file pig.sty.
```

Given such a notice and statement in a file, the conditions given in this license document would apply, with the ‘Work’ referring to the three files ‘pig.dtx’, ‘pig.ins’, and ‘pig.sty’ (the last being generated from ‘pig.dtx’ using ‘pig.ins’), the ‘Base Interpreter’ referring to any ‘LaTeX-Format’, and both ‘Copyright Holder’ and ‘Current Maintainer’ referring to the person ‘M. Y. Name’.

If you do not want the Maintenance section of LPPL to apply to your Work, change ‘main-

tained’ above into ‘author-maintained’. However, we recommend that you use ‘maintained’, as the Maintenance section was added in order to ensure that your Work remains useful to the community even when you can no longer maintain and support it yourself.

Derived Works That Are Not Replacements

Several clauses of the LPPL specify means to provide reliability and stability for the user community. They therefore concern themselves with the case that a Derived Work is intended to be used as a (compatible or incompatible) replacement of the original Work. If this is not the case (e.g., if a few lines of code are reused for a completely different task), then clauses 6b and 6d shall not apply.

Important Recommendations

Defining What Constitutes the Work

The LPPL requires that distributions of the Work contain all the files of the Work. It is therefore important that you provide a way for the licensee to determine which files constitute the Work. This could, for example, be achieved by explicitly listing all the files of the Work near the copyright notice of each file or by using a line such as:

```
% This work consists of all files listed in manifest.txt.
```

in that place. In the absence of an unequivocal list it might be impossible for the licensee to determine what is considered by you to comprise the Work and, in such a case, the licensee would be entitled to make reasonable conjectures as to which files comprise the Work.

B.10 CC0 licence and licence.png licence

The CC0 licence and the licence.png image is licenced under the creative commons attribution international 4.0 licence, this is provided below (NB: This licence is licenced under itself.)

B.11 Creative Commons Attribution 4.0 International Public License

By exercising the Licensed Rights (defined below), You accept and agree to be bound by the terms and conditions of this Creative Commons Attribution 4.0 International Public License ("Public License"). To the extent this Public License may be interpreted as a contract, You are granted the Licensed Rights in consideration of Your acceptance of these terms and conditions, and the Licensor grants You such rights in consideration of benefits the Licensor receives from making the Licensed Material available under these terms and conditions.

B.11.1 Section 1 - Definitions.

- a. **Adapted Material** means material subject to Copyright and Similar Rights that is derived from or based upon the Licensed Material and in which the Licensed Material is translated, altered, arranged, transformed, or otherwise modified in a manner requiring permission under the Copyright and Similar Rights held by the Licensor. For purposes of this Public License, where the Licensed Material is a musical work, performance, or sound recording, Adapted Material is always produced where the Licensed Material is synched in timed relation with a moving image.
- b. **Adapter's License** means the license You apply to Your Copyright and Similar Rights in Your contributions to Adapted Material in accordance with the terms and conditions of this Public License.
- c. **Copyright and Similar Rights** means copyright and/or similar rights closely related to copyright including, without limitation, performance, broadcast, sound recording, and Sui Generis Database Rights, without regard to how the rights are labeled or categorized. For purposes of this Public License, the rights specified in Section 2(b)(1)-(2) are not Copyright and Similar Rights.
- d. **Effective Technological Measures** means those measures that, in the absence of proper authority, may not be circumvented under laws fulfilling obligations under Article 11 of the WIPO Copyright Treaty adopted on December 20, 1996, and/or similar international agreements.
- e. **Exceptions and Limitations** means fair use, fair dealing, and/or any other exception or limitation to Copyright and Similar Rights that applies to Your use of the Licensed Material.
- f. **Licensed Material** means the artistic or literary work, database, or other material to which the Licensor applied this Public License.
- g. **Licensed Rights** means the rights granted to You subject to the terms and conditions of this Public License, which are limited to all Copyright and Similar Rights that apply to Your use of the Licensed Material and that the Licensor has authority to license.
- h. **Licensor** means the individual(s) or entity(ies) granting rights under this Public License.
- i. **Share** means to provide material to the public by any means or process that requires permission under the Licensed Rights, such as reproduction, public display, public performance, distribution, dissemination, communication, or importation, and to make material available to the public including in ways that members of the public may access the material from a place and at a time individually chosen by them.

- j. Sui Generis Database Rights** means rights other than copyright resulting from Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, as amended and/or succeeded, as well as other essentially equivalent rights anywhere in the world.
- k. You** means the individual or entity exercising the Licensed Rights under this Public License. Your has a corresponding meaning.

B.11.2 Section 2 - Scope.

a. License grant .

1. Subject to the terms and conditions of this Public License, the Licensor hereby grants You a worldwide, royalty-free, non-sublicensable, non-exclusive, irrevocable license to exercise the Licensed Rights in the Licensed Material to:
 - A. reproduce and Share the Licensed Material, in whole or in part; and
 - B. produce, reproduce, and Share Adapted Material.
2. **Exceptions and Limitations.** For the avoidance of doubt, where Exceptions and Limitations apply to Your use, this Public License does not apply, and You do not need to comply with its terms and conditions.
3. **Term.** The term of this Public License is specified in Section 6(a).
4. **Media and formats; technical modifications allowed.** The Licensor authorizes You to exercise the Licensed Rights in all media and formats whether now known or hereafter created, and to make technical modifications necessary to do so. The Licensor waives and/or agrees not to assert any right or authority to forbid You from making technical modifications necessary to exercise the Licensed Rights, including technical modifications necessary to circumvent Effective Technological Measures. For purposes of this Public License, simply making modifications authorized by this Section 2(a)(4) never produces Adapted Material.
5. **Downstream recipients. A. Offer from the Licensor - Licensed Material.** Every recipient of the Licensed Material automatically receives an offer from the Licensor to exercise the Licensed Rights under the terms and conditions of this Public License.
 - B. No downstream restrictions.** You may not offer or impose any additional or different terms or conditions on, or apply any Effective Technological Measures to, the Licensed Material if doing so restricts exercise of the Licensed Rights by any recipient of the Licensed Material.

6. No endorsement. Nothing in this Public License constitutes or may be construed as permission to assert or imply that You are, or that Your use of the Licensed Material is, connected with, or sponsored, endorsed, or granted official status by, the Licensor or others designated to receive attribution as provided in Section 3(a)(1)(A)(i).

b. Other rights. 1. Moral rights, such as the right of integrity, are not licensed under this Public License, nor are publicity, privacy, and/or other similar personality rights; however, to the extent possible, the Licensor waives and/or agrees not to assert any such rights held by the Licensor to the limited extent necessary to allow You to exercise the Licensed Rights, but not otherwise.

2. Patent and trademark rights are not licensed under this Public License.

3. To the extent possible, the Licensor waives any right to collect royalties from You for the exercise of the Licensed Rights, whether directly or through a collecting society under any voluntary or waivable statutory or compulsory licensing scheme. In all other cases the Licensor expressly reserves any right to collect such royalties.

B.11.3 Section 3 - License Conditions.

Your exercise of the Licensed Rights is expressly made subject to the following conditions.

a. Attribution .

1. If You Share the Licensed Material (including in modified form), You must:

A. retain the following if it is supplied by the Licensor with the Licensed Material:

i. identification of the creator(s) of the Licensed Material and any others designated to receive attribution, in any reasonable manner requested by the Licensor (including by pseudonym if designated);

ii. a copyright notice;

iii. a notice that refers to this Public License;

iv. a notice that refers to the disclaimer of warranties;

v. a URI or hyperlink to the Licensed Material to the extent reasonably practicable;

B. indicate if You modified the Licensed Material and retain an indication of any previous modifications; and

C. indicate the Licensed Material is licensed under this Public License, and include the text of, or the URI or hyperlink to, this Public License.

2. You may satisfy the conditions in Section 3(a)(1) in any reasonable manner based on the medium, means, and context in which You Share the Licensed Material. For example, it

may be reasonable to satisfy the conditions by providing a URI or hyperlink to a resource that includes the required information.

3. If requested by the Licensor, You must remove any of the information required by Section 3(a)(1)(A) to the extent reasonably practicable.
4. If You Share Adapted Material You produce, the Adapter's License You apply must not prevent recipients of the Adapted Material from complying with this Public License.

B.11.4 Section 4 - Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed Material:

- a. for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, and Share all or a substantial portion of the contents of the database;
- b. if You include all or a substantial portion of the database contents in a database in which You have Sui Generis Database Rights, then the database in which You have Sui Generis Database Rights (but not its individual contents) is Adapted Material; and
- c. You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of the contents of the database.

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under this Public License where the Licensed Rights include other Copyright and Similar Rights.

B.11.5 Section 5 - Disclaimer of Warranties and Limitation of Liability.

- a. Unless otherwise separately undertaken by the Licensor, to the extent possible, the Licensor offers the Licensed Material as-is and as-available, and makes no representations or warranties of any kind concerning the Licensed Material, whether express, implied, statutory, or other. This includes, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of latent or other defects, accuracy, or the presence or absence of errors, whether or not known or discoverable. Where disclaimers of warranties are not allowed in full or in part, this disclaimer may not apply to You.
- b. To the extent possible, in no event will the Licensor be liable to You on any legal theory (including, without limitation, negligence) or otherwise for any direct, special, indirect, incidental, consequential, punitive, exemplary, or other losses, costs, expenses, or damages arising out of this Public License or use of the Licensed Material, even if the Licensor has been advised of the possibility of such losses, costs, expenses, or damages. Where a limitation of liability is not allowed in full or in part, this limitation may not apply to You.

- c. The disclaimer of warranties and limitation of liability provided above shall be interpreted in a manner that, to the extent possible, most closely approximates an absolute disclaimer and waiver of all liability.

B.11.6 Section 6 - Term and Termination.

- a. This Public License applies for the term of the Copyright and Similar Rights licensed here. However, if You fail to comply with this Public License, then Your rights under this Public License terminate automatically.
- b. Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:
 - 1. automatically as of the date the violation is cured, provided it is cured within 30 days of Your discovery of the violation; or
 - 2. upon express reinstatement by the Licensor.

For the avoidance of doubt, this Section 6(b) does not affect any right the Licensor may have to seek remedies for Your violations of this Public License.

- c. For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate terms or conditions or stop distributing the Licensed Material at any time; however, doing so will not terminate this Public License.
- d. Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

B.11.7 Section 7 - Other Terms and Conditions.

- a. The Licensor shall not be bound by any additional or different terms or conditions communicated by You unless expressly agreed.
- b. Any arrangements, understandings, or agreements regarding the Licensed Material not stated herein are separate from and independent of the terms and conditions of this Public License.

B.11.8 Section 8 - Interpretation.

- a. For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduce, limit, restrict, or impose conditions on any use of the Licensed Material that could lawfully be made without permission under this Public License.
- b. To the extent possible, if any provision of this Public License is deemed unenforceable, it shall be automatically reformed to the minimum extent necessary to make it enforceable. If the

provision cannot be reformed, it shall be severed from this Public License without affecting the enforceability of the remaining terms and conditions.

- c. No term or condition of this Public License will be waived and no failure to comply consented to unless expressly agreed to by the Licensor.
- d. Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver of, any privileges and immunities that apply to the Licensor or You, including from the legal processes of any jurisdiction or authority.

B.12 ulem.sty licence

Copyright (c) 1989-2011 by Donald Arseneau (Vancouver, Canada; asnd@triumf.ca)

This software may be freely transmitted, reproduced, or modified for any purpose provided that this copyright notice is left intact. Small excerpts may be taken and used without any restriction.)

B.13 turtles.ttf Licence

don't have
licence, see
: //www.pixelsagas.com/

B.14 tikz-uml.sty licence

B.15 tikz-styles.sty licence