Rapport Google Wave for Research and Education

Project:	SURFworks / Google Wave
Projectjaar:	2009
Projectmanager:	F.Pinxt
Auteur(s):	Peter Clijsters (Everett), Paul van Dijk (SURFnet), Niels van Dijk (SURFnet)
Opleverdatum:	13 oktober 2009
Versie:	1.0 definitief

Samenvatting

Deze technologieverkenning onderzoekt allereerst de Wave technologie op basis van de beschikbare documentatie en inventariseert de mogelijkheden de SURFfederatie te gebruiken om in autorisatie en authenticatie te voorzien. Hierbij wordt gekeken naar zowel de technische als de functionele aspecten.
Het rapport beschrijft de Wave technologie op basis van de beschikbare documentatie en inventariseert de mogelijkheden de SURFfederatie te gebruiken om in autorisatie en authenticatie te voorzien en doet daarnaast een aantal aanbevelingen over implementatie en opzet en wellicht aanvullingen op het protocol.

Voor deze publicatie geldt de Creative Commons Licentie “Attribution-Noncommercial-Share Alike 3.0 Netherlands”.
Meer informatie over deze licentie is te vinden op http://creativecommons.org/licenses/by-nc-sa/3.0/nl/

Colofon

Programmalijn	: SURFworks Technologieverkenning Google Wave
Onderdeel	: Rapport Google Wave for research and Education
Activiteit	: geen nummer in jaarplan
Deliverable	: geen benaming in jaarplan
Toegangsrechten	: publiek
Externe partij	: Everett

Dit project is tot stand gekomen met steun van SURF, de organisatie die ICT vernieuwingen in het hoger onderwijs en onderzoek initieert, regisseert en stimuleert door onder meer het financieren van projecten. Meer informatie over SURF is te vinden op de website (www.surf.nl).

4 dingen die je moet weten over het Rapport Google Wave for Research and Education	
Context	Het Google Wave platform en met name de onderliggende Wave API's lijken een interessante nieuwe wijze van samenwerken mogelijk te maken. Het feit dat Google voornemens is zowel de API's als ook de software voor cliënt en server in een open licentie beschikbaar te maken, maakt het voor SURFnet en haar instellingen extra interessant. Wel zal natuurlijk de vraag spelen of en op welke wijze deze technologie kan samenwerken met de SURFfederatie.
Wat is het?	Het rapport geeft een technische en functionele beschrijving van de Google Wave technologie.
Voor wie is het?	Het rapport “Google Wave for Research and Education” is bestemd voor technisch georiënteerde lezers die geïnteresseerd zijn in het onderwerp en voor het Programma Management SURFworks.
Hoe werkt het?	Google Wave is een nieuw veelbelovend communicatieplatform van Google waarbij email, instant messaging, conversatie en bewerken van documenten in elkaar over vloeien.
Wat kan je ermee?	Google Wave biedt een nieuwe manier van communicatie en onderlinge samenwerking.
Extra (Bijlagen, Thema, Gerelateerde thema’s)	Rapport Google Wave for Research and Education

GOOGLE WAVE
TECHNOLOGY SCOUTING

A FIRST LOOK AT GOOGLE WAVE FROM THE NREN PERSPECTIVE

version 1.0, september 2009

SURFnet bv, Radboudkwartier 273, Postbus 19035, 3501 DA Utrecht
T +31 302 305 305, F +31 302 305 329, www.SURFnet.nl

CONTENT

1 Introduction 3
1.1 Background 3
1.2 Project goals 4
1.3 Reading guide 4
2 Summary of Google Wave protocol 5
2.1 Introduction 5
2.2 Wave Providers 6
2.3 Wave entities 7
2.4 Wave Service Architecture 9
2.5 Federation Host and Federation Remote 10
2.6 Protocol and security 10
3 Google Wave APIs 12
3.1 Extensions 12
3.2 Embed API 13
4 Authentication and Authorisation 15
4.1 Accessing a Wave Provider 15
4.2 Backend authentication 16
4.3 Authorisation 20
5 Wave for Education and eScience 21
5.1 Introduction 21
5.2 Wave for Education 22
5.3 Wave for eScience 23
6 Conclusions and recommendations 27

1 INTRODUCTION

1.1 Background

On May 28, 2009 at Google I/O, Google unveiled a developer preview of Google Wave, a new collaboration and communication product. Google Wave introduces a new platform built around hosted conversations called waves - this model enables people to communicate and work together in new and more effective ways. On top of that, with the Google Wave APIs, developers can take advantage of this collaborative system by building on the Google Wave platform. Google wants to expand upon that platform, which is why they've put together the initial draft of the Google Wave Federation Protocol, the underlying network protocol for sharing waves between wave providers.

As the protocol is open source, anyone can build a wave server and interoperate, much like anyone can run their own SMTP server. The wave protocol is open to contributions by the broader community with the goal to continue to improve how we share information, together.

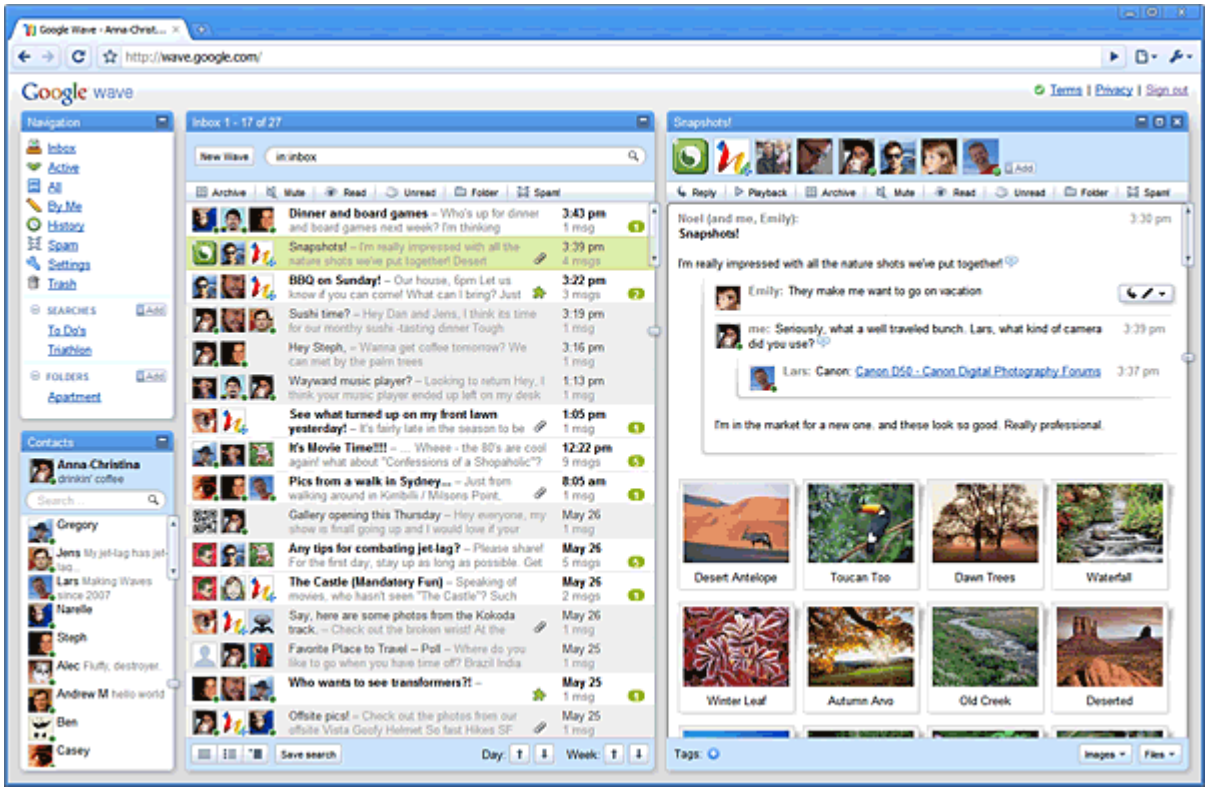


Figure 1 Impression of Google Wave client interface

1.2 Project goals

The concept of Google Wave and especially the underlying platform APIs promise interesting new ways of cooperation. The fact Google plans to release both API and the software for client and server in an open license would allow SURFnet and its institutions to freely use this technology, thus making it even more interesting.

This Technology Scouting investigates Google Wave technology based on the available documentation. It will consider both the technical and functional aspects of Google Wave. One point of interest will be the possibility for use with SURFfederatie for providing authentication and authorization. Furthermore it will identify opportunities for usage within the NREN (National research and Education Network) communities. Finally it will make recommendations for further investigation or Proof of Concept project.

1.3 Reading guide

The first chapter of this report will dive into the concepts behind Google Wave. Blibs, Waves and wavelets will be introduced as well as the way Google Wave handles client-server and server-server communication. The next chapter introduces the Google Wave API and shows how robots and gadgets can interact with the wave. This is followed by a chapter that looks into authorisation and authentication within Google Wave, and the possibilities of using SURFfederatie for handling this. In chapter 5 use cases for Google Wave are identified. This chapter points out the already available services, and describes two use cases for using Google Wave in both an educational and escience context. The document is concluded with a chapter describing the authors' conclusions and recommendations.

2 SUMMARY OF GOOGLE WAVE PROTOCOL

The Google Wave protocol is currently being developed and the draft specifications[1] are published on http://www.waveprotocol.org. These specifications are mutated regularly as progress on the protocol is being made. Some major protocol capabilities have not yet been defined, like authorisations for mutating wavelets and administration of groups. This makes creating a full picture of the wave universe difficult. This chapter discusses the protocol capabilities that have been defined so far and makes use of documentation present on waveprotocol.org.

2.1 Introduction

The Google Wave Federation Protocol Over XMPP is an open extension to the XMPP core protocol[2] allowing near real-time communication of wave updates between two wave servers. Note that in the context of Google Wave, the term “Federation” relates to the fact that Google Wave communication is performed between separate domains and that waves are shared among users from these domains. It is not related to the concept of Identity Federation which is implemented by the SURFfederatie.

The Google Wave architecture is a communication and collaboration platform based on hosted conversations, called waves. A wave consists of XML documents and supports concurrent modifications and low-latency updates between participants on the wave.

Figure 2 gives an overview of the components involved in the Google Wave architecture.

---

[1] Google Wave Federation Protocol Over XMPP, 8 June 2009, http://www.waveprotocol.org/draft-protocol-spec

[2] RFC 3920 - Extensible Messaging and Presence Protocol (XMPP): Core, Saint-Andre, P., Ed., October 2004, http://tools.ietf.org/html/rfc3920

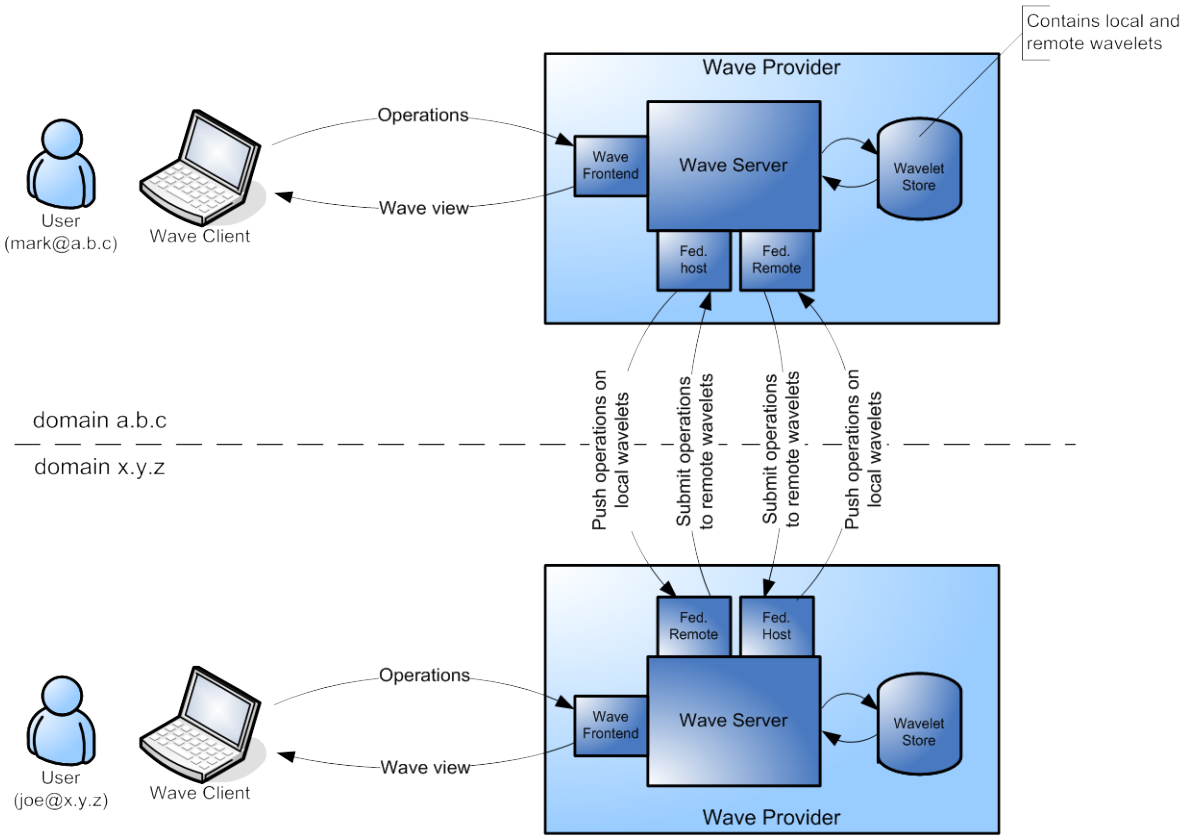


Figure 2 Google Wave architecture

2.2 Wave Providers

The wave federation protocol enables everyone to become a wave provider and share waves with others. For instance, an organization can operate as a wave provider for its members, an individual can run a wave server as a wave provider for a single user or family members, and an Internet service provider can run a wave service as another Internet service for its users as a supplement to email, IM, ftp, etc. In this model, Google Wave is one of many wave providers.

A wave provider is identified by its Internet domain name(s).

Wave users have wave addresses which consist of a user name and a wave provider domain in the same form as an email address, namely <username>@<domain>. Wave addresses can also refer to groups, robots, gateways, and other services. A group address refers to a collection of wave addresses, much like an email mailing list. A robot is an automated participant on a wave (see the robots API). Examples are translation robots and chess game robots. A gateway translates between waves and other communication and sharing protocols such as email and IM. In the remainder we ignore addressees that are services, including robots and gateways - they are treated largely the same as users with respect to federation.

Wave users access all waves through their wave provider. If a wave has participants from different wave providers, their wave providers all maintain a copy of the wave and serve it to their users on the wave. The wave providers share updates to the wave with each other using the wave federation protocol which we describe below. For any given wave user, it is the responsibility of the wave provider for the user's domain to authenticate the user (using cookies and passwords, etc) and perform local access control.

2.3 Wave entities

A wave consists of the following entities (see Figure 3):

- A **wave** can be viewed as a threaded conversation consisting of one or more participants (which may include both human participants and robots). A wave serves as a container for one or more wavelets.
- A **wavelet** is a threaded conversation that is spawned from a wave (including the initial conversation). Wavelets serve as the container for one or more documents.
- A **document** can represent the rich-text messages (known as “blips”) that make up a conversation in the wavelet or can represent other data not displayed to the user as part of the threaded conversation.

These entities are described in more detail in the next sub-paragraphs.

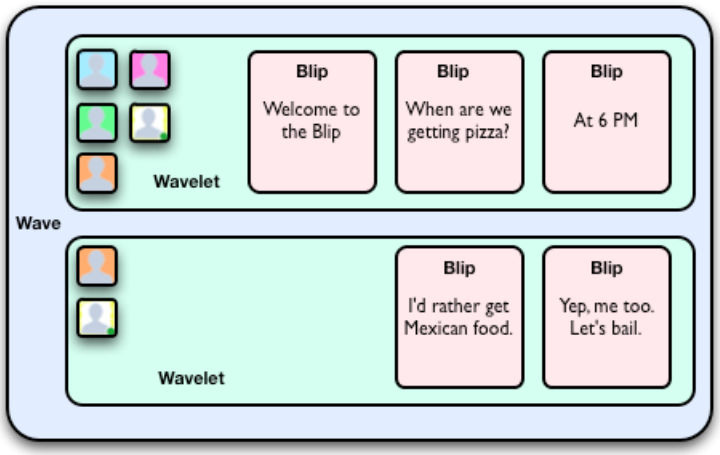


Figure 3 Wave entities overview

2.3.1 Waves, Wavelets, and Identifiers

A wave consists of a set of wavelets. When a user has access to a wavelet, that user is called a participant of that wavelet. Each wavelet has a list of participants, and a set of documents that make up its contents. Different wavelets of a wave can have different lists of participants. Copies of a wavelet are shared across all of the wave providers that have at least one participant

in that wavelet. Amongst these wave providers, there is a designated wave provider that has the definitive copy of that wavelet. We say that this particular provider is hosting that wavelet.

When a user opens a wave, a view of the wave is retrieved, namely the set of wavelets in the wave that the user is a participant of (directly, or indirectly via group membership). In general, different users have different wave views for a given wave. For example, per-user data for a user in a wave, such as the user's read/unread state for the wave, is stored in a user-data wavelet in the wave with the user as the only participant. The user-data wavelet only appears in this user's wave view. Another example is a private reply within a wave, which is represented as a wavelet with a restricted participant list. The private reply wavelet is only in the wave views of the restricted list of users.

A wave is identified by a globally unique wave id, which is a pair of a domain name and an id string. The domain names the wave provider where the wave originated.

A wavelet has a wavelet id which is unique within its wave. Like a wave id, a wavelet id is a pair of a domain name and an id string. The domain name in the wavelet id plays a special role: It names the wave provider that hosts the wavelet. A wavelet is hosted by the wave provider of the participant who creates the wavelet. The wave provider who hosts a wavelet is responsible both for operational transformation and application of wavelet operations to the wavelet and for sharing the updates with the wave providers of all the wavelet participants, as described in the Wave Servers section below. The updates are wavelet operations and concurrent updates are resolved using operational transformation.

Wavelets in the same wave can be hosted by different wave providers. For example, a user-data wavelet is always hosted by the user's wave provider, regardless of where the rest of the wave is hosted. Indeed, user-data is not federated, i.e., not shared with other wave providers. Another example is a private reply wavelet. A particularly simple instance of this is when all the participants of the private reply are from the same wave provider. Then this wave provider will not share the private reply wavelet with other wave providers, regardless of where the other wavelets in the wave are hosted.

2.3.2 Documents and blips

Each wavelet is a container for any number of documents. Each document has an id that is unique within its containing wavelet. It is composed of an XML document and a set of annotations. Annotations are key-value pairs that span arbitrary ranges of the XML document and are independent of the XML document structure. They are used to represent text formatting, spelling suggestions and hyper-links.

Some documents represent rich-text messages in the wavelet. These are known as "blips". The blips in a wave form a threaded conversation. Other documents represent data, for example tags, and are not displayed to the user as part of the threaded conversation structure in the wave. More detailed

information on the structure of documents can be found in the Google Wave Operational Transformation white paper[3].

Documents can also contain embedded binary attachments, such as images, PDF documents and ZIP archives.

2.4 Wave Service Architecture

A wave provider operates a wave service on one or more networked servers. The central pieces of the wave service is the wave store, which stores wavelet operations, and the wave server, which resolves wavelet operations by operational transformation and writes and reads wavelet operations to and from the wave store. Typically, the wave service serves waves to users of the wave provider which connect to the wave service frontend (see "Google Wave Data Model and Client-Server Protocol"), and we shall assume this in the following description of the wave service architecture. More importantly, for the purpose of federation, the wave service shares waves with participants from other providers by communicating with these wave provider's servers.

For a given wave provider, its wave server serves wave views to local participants, i.e., participants from its domain. This wave server stores the state of all wavelets that it knows about. Some are hosted by the wave server itself. These are "local wavelets" relative to this wave server. Others are copies of wavelets hosted by other wave providers. These are "remote". A wave view can contain both types of wavelet simultaneously.

At a particular wave provider, local wavelets are those created at that provider, namely by users who belong to the wavelet provider. The wave server is responsible for processing the wavelet operations submitted to the wavelet by local participants and by remote participants from other wave providers. The wave server performs concurrency control by ordering the submitted wavelet operations relative to each other using operational transformation[4]. It also validates the operations before applying them to a local wavelet.

Remote wavelets are hosted by other wave providers. The wave server maintains cached copies locally and updates them with wavelet operations that it gets from the hosting wave providers. When a local participant submits a wavelet operation to a remote wavelet, the wave server forwards the operation to the wave server of the hosting provider. When the transformed and applied operation is echoed back, it is applied to the cached copy. Read access to local participants is done from the cached copy without a round trip to the hosting wave provider.

---

[3] Google Wave Operational Transformation, David Wang, Alex Mah, http://www.waveprotocol.org/whitepapers/operational-transform

[4] General Verifiable Federation, Lea Kissner, Ben Laurie, 27 May 2009, http://www.waveprotocol.org/whitepapers/wave-protocol-verification

Local and remote wavelets are all stored in the wave server's persistent wave store.

We say that a wave provider is "upstream" relative to its local wavelets and that it is "downstream" relative to its remote wavelets.

2.5 Federation Host and Federation Remote

The wave service uses two components for peering with other wave providers, a "federation host" and a "federation remote"[5].

The federation host communicates local wavelet operations, i.e., operations on local wavelets:

- It pushes new wavelet operations that are applied to a local wavelet to the wave providers of any remote participants.
- It satisfies requests for old wavelet operations.
- It processes wavelet operations submission requests.

The federation remote communicates remote wavelet operations and is the component of a wave provider that communicates with the federation host of upstream wave providers:

- It receives new wavelet operations pushed to it from the wave providers that host the wavelets.
- It requests old wavelet operations from the hosting wave providers.
- It submits wavelet operations to the hosting wave providers.

An upstream wave provider's federation host connects to a downstream wave provider's federation remote to push wavelet operations that are hosted by the upstream wave provider.

2.6 Protocol and security

The network protocol between federation host and remote is called the Google Wave Federation Protocol. It is implemented as an open extension to the XMPP Internet Messaging protocol. It uses XEP-0114[6], the Jabber Component Protocol, which is a standard way for an XMPP server to support extensions (see Figure 4). Some key useful features of XMPP that the wave federation protocol uses are the discovery of IP addresses and ports, using SRV records, and TLS authentication and encryption of connections.

---

[5] In an earlier revision of this draft specification these components were called "federation gateway" and "federation proxy", respectively

[6] XEP-0114: Jabber Component Protocol, Peter Saint-Andre, version 1.5, 03 March 2005, http://xmpp.org/extensions/xep-0114.html

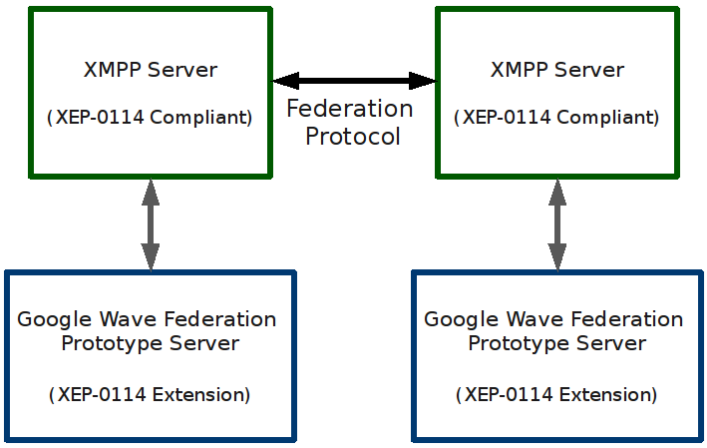


Figure 4 Google Wave server as an XMPP extension

The XMPP transport encrypts operations at a transport level, so it only provides cryptographic security between servers that connect directly to each other. An additional layer of cryptography provides end-to-end authentication between wave providers using cryptographic signatures and certificates, allowing all wavelet providers to verify the properties of the operation. Specifically, a downstream wave provider can verify that the wave provider is not spoofing wavelet operations, namely, it cannot falsely claim (1) that a wavelet operation originated from a user on another wave provider or (2) that it was originated in a different context. This addresses the situation where two users from different, trustworthy wave providers, say love.com and peace.com, are participants of a wavelet that is hosted on a malicious wave provider evil.com. The protocol requires love.com to sign its user's operations with love.com's certificate and peace.com to sign its user's operations with peace.com's certificate. These signatures travel with the operations and evil.com must host the signatures together with the operations. Furthermore, love.com and peace.com will verify the signatures of all the operations that evil.com forwards. This makes it impossible for evil.com to alter or spoof the contents of the messages from the user of love.com which is shared with peace.com, and vice versa. All the signing and verification is done by the wave providers, not the client software of the end users.

The protocol specification requires that wave providers connecting using the federation protocol must authenticate using cryptographically secure TLS mechanisms. Moreover, it is recommended that they use TLS to encrypt the traffic between them. The client-server and federation protocols do not provide end-to-end authentication or encryption between end users. A wave provider should authenticate its end users and it is recommended that it offers its end users secure encrypted connections to its wave service frontend. In combination, secure connections between wave services and secure connections between users and their wave services offer a reasonable level of security between end clients.

3 GOOGLE WAVE APIS

Google Wave's reach and functionality can be extended by developing new wave components or embedding waves in other systems[7].

- Developers can enhance Google Wave itself by authoring mini-applications (**extensions**) that interact with the Wave. Developers can create robots which interact with waves, or gadgets which participants may interact with (and combinations of robots and gadgets are also supported).
- Developers can enhance their existing web applications by **embedding** Google Wave directly in their application, allowing them to seamlessly integrate communication and collaboration.

3.1 Extensions

Wave extensions are a way to augment the functionality of a wave[8]. Currently, the Wave API supports robots and gadgets extensions:

- Robot extensions automate common tasks. Robots are automated wave participants backed by software rather than human beings. Robots are applications which run in the "cloud" and can modify state within the wave itself. A robot can read the contents of a wave in which it participates, modify the wave's contents, add or remove participants, and create new blips and new waves. Robots perform actions in response to events. For example, a robot might publish the contents of a wave to a public blog site and update the wave with user comments.
- Gadget extensions provide a new way for users to interact. Gadgets provide a means to run small programs in waves. A gadget is a small application that runs within a client. The gadget is owned by the wave, and all participants on a wave share the same gadget state. The only events a gadget responds to are changes to its own state object, and changes in the wave's participants (for example, participants joining or leaving the wave). The gadget has no influence over the wave itself. Wave gadgets typically aren't full blown applications, but small add-ons that improve certain types of conversations. For example, a wave might include a sudoku gadget that lets the wave participants compete to see who can solve the puzzle first.

This table summarizes the differences:

---

[7] Google Wave API Overview, http://code.google.com/apis/wave/guide.html

[8] For more information, see "Wave Extensions", http://code.google.com/apis/wave/extensions/

Robot	Gadget
Runs in the cloud. Currently, only Google App Engine is supported. In the future, any client architecture that implements the Wave protocol will be supported.	Runs on the client.
One instance of a given robot per wave. Remember that a robot is like a participant on a wave, so each participant/robot is a unique instance, but a wave can have many participants/robots.	Multiple instances per wave.
Able to modify a wave and perform the same operations as a human participant.	Not able to modify the wave, and has limited visibility into the wave. Is only able to detect changes in the wave's participants.
Can modify a gadget.	Unable to modify a robot, has no way to know that the robot exists.

3.2 Embed API

The Embed API[9] allows third party websites anywhere on the web to embed waves for easy discussion and collaboration using wave's user interface. An embedded wave acts and behaves the same as a wave within the Google Wave client application. An embedded wave even supports drag-and-drop functionality, allowing the web application to provide users with an interactive collaboration tool.

[9] Wave Embed Developer's Guide, http://code.google.com/apis/wave/embed/

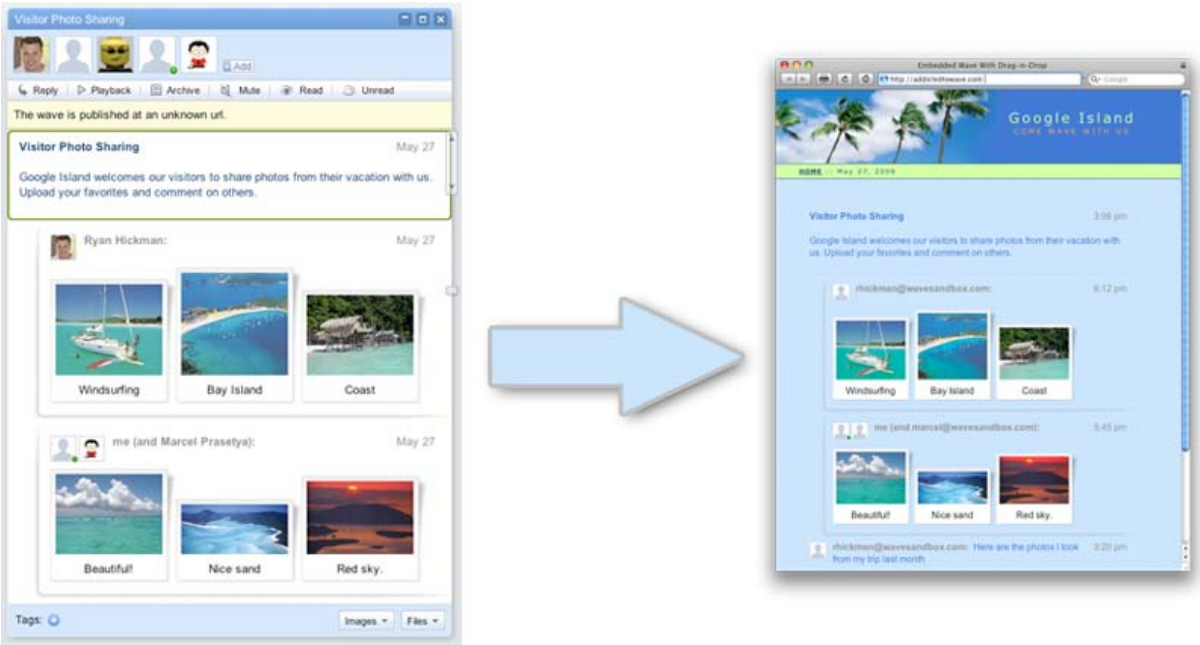


Figure 5 Embedded wave

4 AUTHENTICATION AND AUTHORISATION

Of special interest to SURFnet is the authentication and authorisation methods supported by Google Wave. At several points in the wave architecture, authentication and authorisation play an important role:

- Communication between wave servers. This is discussed in paragraph 2.6.
- A user accessing a wave provider
- A wave provider accessing a backend service provider

The last two bullets are discussed in this chapter.

4.1 Accessing a Wave Provider

An important concept in Google Wave is that a user belongs to a particular wave provider. A user (or maybe “account” would be a more accurate term) only uses this specific wave provider to access the wave universe. Therefore, a wave provider only has to support known users it can authenticate.

When looking at the current state of the wave protocol specification and released whitepapers, authenticating and accessing a Wave Provider is not specified. In fact, in the specification it is stated that:

> “For any given wave user, it is the responsibility of the wave provider for the user's domain to authenticate the user (using cookies and passwords, etc) and perform local access control.”

Not more elaboration is given as to how the provider should perform authentication. It is therefore assumed that it is up to a specific wave provider implementation to support certain authentication methods. These methods can range from username/password, X.509 certificates to software or hardware tokens but could also include identity federation techniques like the ones employed by the SURFfederatie.

For SURFnet, leveraging the SURFfederatie for access to wave providers would be the most preferred option. When SURFnet would itself become a wave provider and use the SURFfederatie for authentication, potentially all users from the connected institutions could make use of this service. Initially, these users would be unknown to the wave provider, but some mechanism could be implemented that would create a wave provider account on-the-fly after a user login. Possibly asking some first-time questions like setting preferences and accepting a use policy.

Other institutions connected to SURFnet could employ the same mechanism, possibly limiting the users their wave provider supports to only students and employees of that specific institution. They could use their institutions LDAP to perform the authentication, but it would make sense to use the general SURFfederatie and limiting it to their own Identity Provider. The reason for this is that with using the SURFfederatie for the front-end authentication,

Single Sign-On (SSO) to backend services can be achieved. This is discussed in the next paragraph.

4.2 Backend authentication

One of the most relevant features of Google Wave for SURFnet would be the integration with other collaboration services like document management systems, wiki's, e-learning tools, bug tracking tools, etc. In these types of scenarios, typically a wave robot handles the communication between the wave and the service. Messages posted in a wave with such a robot attached can be transported to the service and vice-versa, appearing instantly in both systems. Examples of these types of robots already exist, there is Tweety for Twitter and Bloggy for Blogger sites.

To be able to publish anything to these kind of collaboration services usually requires authentication. This is also true for a wave robot sending posts to a service. Initially, the robots demoed by Google used an authentication scheme, which involved passing a user's username and password in plain-text around wave. This offered very little security and has now been succeeded by OAuth[10].

The OAuth use case seems to be a very good fit for usage by Google Wave (see paragraph 4.2.1 and 4.2.2). An other technique that could be used for backend authentication has previously been researched by SURFnet[11] and uses the WS-Trust standard. Paragraph 4.2.3 summarizes this and describes how this could be used by Google Wave.

4.2.1 OAuth

The OAuth use case is shown in Figure 6. An end user uses a web application (Google Wave in this case) hosted by the Consumer (the Wave Provider). At some point, the Consumer needs to access a service hosted by the Service Provider. This service requires an end user authentication. The OAuth authentication scheme is described below.

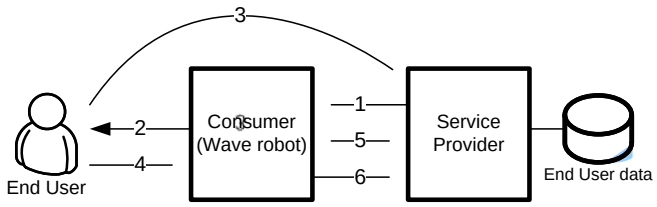


Figure 6 OAuth Wave use case

[10] OAuth: An open protocol to allow secure API authorization in a simple and standard method from desktop and web applications. http://oauth.net/

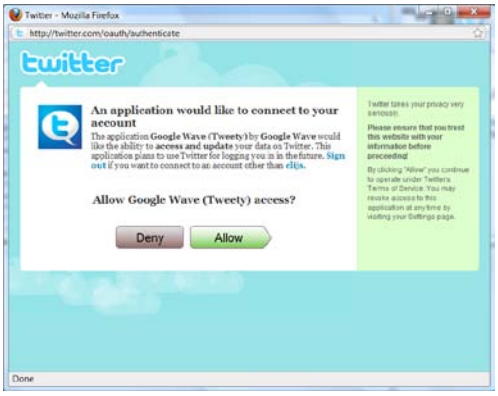
[11] Webservices en SURFfederatie, Peter Clijsters, 28/05/2009

1. The Consumer requests a Request Token from the SP
2. The Request Token is given to the end user
3. The end user authorizes the Request Token by authenticating himself to the SP. After authentication, the user must authorize the SP to allow the Consumer to access the service in the name of the end user. Optionally the user can set the access requirements for the Consumer (eg what information the Consumer can request from the SP and the validity period for the access)
4. The end user gives the authorized Request Token to the Consumer
5. The Consumer exchanges the authorized Request Token at the SP for an Access Token
6. With this Access Token, the Consumer can access the service hosted by the SP.

For all communications, OAuth uses HTTP(S) whereby the OAuth specific parameters are communicated using HTTP headers and URL parameters.

When applied to Google Wave, the Consumer is actually a wave robot hosted on a wave server. Depending on specific access requirements from the Service Provider, the robot can store the authorized Request Token for a period of time. This would eliminate the need for asking the end user to authenticate every time the robot needs to access the service.

An example[12] of a wave robot using OAuth for authentication is Tweety. This robot posts entries in a wave to a Twitter account and vice-versa. The first time this robot is added to a wave, the user needs authenticate to Twitter. The screenshots for authentication are shown below.



After the user performed a successful authentication and allows Tweety to access the Twitter account in the users name, the robot starts working. The access token generated by this initial OAuth authentication is stored by

[12] http://googlewavedev.blogspot.com/2009/08/tweety-sample-improved-with-oauth.html#links

Tweety so it can be reused, eliminating the need for authentication for subsequent interactions between Tweety and Twitter. Note that this access token will still adhere to Twitters access requirements.

4.2.2 OAuth and SURFfederatie login

The OAuth use case in the previous paragraph assumes the Service Provider can authenticate the end user. In the SURFnet environment, most Service Providers will not be able to authenticate end users directly but will outsource this to the SURFfederatie. The OAuth specification does not specify the method of authentication used by the Service Provider. Using identity federation for this purpose is possible and OpenID for example is already used regularly with OAuth. Figure 7 shows the additional authentication steps (3a and 3b) necessary when using the SURFfederatie with OAuth in a Google Wave context.

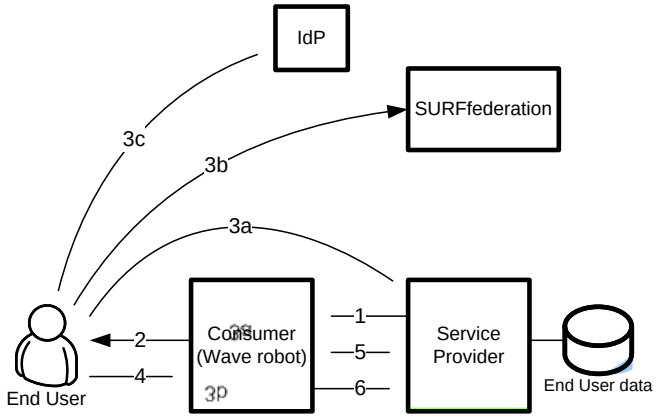
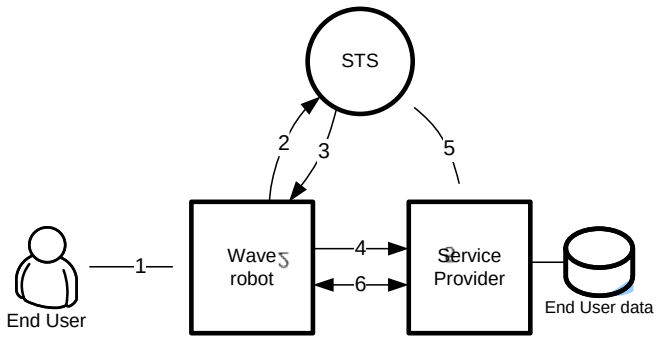


Figure 7 OAuth Wave use case with SURFfederatie

When the initial access to Google Wave (see paragraph 4.1) was already authenticated by the SURFfederatie, the OAuth SURFfederatie authentication step (step 3b) has already been performed. This enables Single Sign-On (SSO) between Google Wave and backend Service Providers when using the SURFfederatie.

4.2.3 WS-Trust



1. When using Google Wave, the end user requests a service from a wave robot.

2. The robot sends a Requests Security Token (RST) message to the WS-Trust Security Token Service (STS), requesting a security token. With this request, the robot needs to pass some sort of proof-of-identity the STS can validate. This proof-of-identity can be a username/password, X.509 certificate, SAML or Kerberos token. Note that in a diverse environment, being able to validate this token is not an easy task for the STS.

3. After validating the token in the RST message, the STS creates the requested security token and sends it back to the robot in a Request Security Token Response (RSTR) message.

4. The wave robot embeds the received security token into the request sent to the Service Provider.

5. The Service Provider validates the security token with the STS. This can be done by using a validation binding to the STS or by establishing a trust between the Service Provider and the STS beforehand. After validation, the Service Provider processes the request from the robot and provides a response. Also, this is the moment where a security context should be created between the robot and the Service Provider, for example by using a cookie and session mechanism or possibly by using WS-SecureConversation standard.

6. Subsequent requests from the robot to the Service Provider can take place in the security context setup by the previous step. This eliminates the need to perform steps 2-5 each time the robot needs to access the Service Provider.

WS-Trust uses SOAP messages to communicate with the STS. Furthermore, communication between the robot and the Service Provider needs to be in the form of SOAP messages.

4.3 Authorisation

Authorisation in Google Wave is currently not very developed. The access control to a wavelet is simply boolean, that is, a user can either have full read-write access to a wavelet, or have no access to a wavelet. Google is discussing adding more fine grained access control, but has not yet provided a proposal[13].

[13] http://groups.google.com/group/wave-protocol/browse_thread/thread/4fb0a32e14efc1ee

5 WAVE FOR EDUCATION AND ESCIENCE

5.1 Introduction

Will Google Wave be the Swiss Army Knife it is intended to be? Will it redefine the way researchers, educators and students collaborate? The answer to this question largely depends on the extensions (robots) that will become available for Google Wave.

Right now, robot extensions are mostly focussing on funny applications - we are still at the playing stage. The real endeavour is to invent and implement robots that serve research and educational needs. In a SURFnet setting this could be by providing SURFnet federated access to a wave server and providing SURFnet services, institutional and commercial services in a wave.

With the standard wave client a number of interesting features are available already:

- Collaboration on documents
- Playback of actions within a wave
- Links to other waves
- Tags in waves
- Searching within waves

With robot extensions many additional features may become available:

- Accessing repositories (e.g. with the Robot Igor)
    - Protein databases (omit)
    - Reference libraries (pubmed,science direct)
- Accessing document repositories from within a wave
- Initiating webconferencing sessions from within a wave
- Select participants for wave based on social distance in Social Networks (LinkedIn)
- Sharing bookmarks (social bookmarking)
- Identification of funding sources Personal networks; funding agencies; institutional resources e.g. SAMI
- Scientific workflow and project management
- Collaborative data-analyses with a wave client as the front-end
- Everything you write can be canalized to different places: blogs, wikis, document repositories, email.

With the basic features and these extensions, Google Wave may:

- Change the way we draft scientific manuscripts
- Support the analysis and publishing processes
- Help in battling information overload
- Improve collaboration in a classroom setting

5.2 Google Wave for Education

In an educational context one can think of applying the Google Wave concept on many different levels (institutional – faculty – classroom – internships..). The use case below illustrates how Google Wave can be used for a marketing course.

5.2.1 A small marketing course

The following is used with permission from Victor Maijer's blog[14]

Let's say I have a small Marketing course in which I want to touch two topics, Product Mix and Branding. I give two lectures on Product Mix and Branding.

I would like to explore how this would look in a Blackboard or Sakai environment.

I would use "My Documents" (Blackboard) or Resources (Sakai) and make two sub folders (Product Mix and Branding). The slides of the lectures would end up in the sub folders and would add some additional links or such in the sub folders. When I uploaded the slides of particular lecture I will make an announcement that the slides are in place. Furthermore, I would use Announcements for general, you guessed it, announcements. And maybe I'll activate a discussion board. It is pretty hard to have a good solid discussion board since it requires guidance and some incentives for students in order to function. At least students are able to drop some questions.

How would I do this in Google Wave? This is where the fun starts.

In Google Wave, I would start a wave. Not just one but multiple waves. I tend to see a wave within a learning context as an 'hosted learning conversation'.

My Branding lecture would be a wave. An announcement, document, and discussion are blended into one single wave. This enables me to focus on the subject. I don't have to worry which tools I should enable and apply for this course.

What about your Product Mix lecture? Well, that's going to be a separate wave, just like my Branding wave. And what do you do with general announcements? Separate wave. This can become quite messy so we have to organize ourselves. I would do two things to get some order. First I would start tagging all the Marketing waves. In this example that would mean that my "Branding", "Product Mix", and "General Announcement" wave all are tagged as "Marketing".

The second thing I would do is to create a "Master Marketing" wave. Google Wave enables you to link to other waves, which creates a Wiki-ish structure. So I from my Master wave I can link to other waves like my Branding wave.

---

[14] http://victormaijer.blogspot.com/2009/05/google-wave-education-simple-use-case.html

So I and up with a wave structure that supports my marketing course and enables me to have an 'hosted learning conversation' with my students.

5.3 Google Wave for eScience

How does Google Wave fit in the need for online collaboration in research? It is reasonably clear that there are two short to medium term applications for Google Wave in the research routine. The first is the collaborative authoring of documents and the discussions around those. The second is a more challenging application: the use of Google Wave as a recording and analysis platform. The latter heavily depends on the availability of wave robots and requires that current analysis tools (best of breed) have interfaces/APIs available that enable the use of robots on those environments.

Research is an iterative and non-linear process and contains many of the activities shown in the figure below. In each of these steps Google Wave could play a role.

5.3.1 A simple use case: Alzheimer disease

Hypothesis

In the hypothesis generation phase, one could think of discussions and threads on a particular topic which can be supported by the standard wave

client. For example, a small group of scientists and physicians from University X are studying the impact of certain genes on the development and progression of Alzheimer disease. There is a strong indication that a mutation in the ABC gene plays an important role in the onset of Alzheimer and they all believe that further study to this gene is worthwhile.

As most staff members are also involved in clinical work, it seems difficult to keep the pace in this project. Therefore they decide to start a wave on gene ABC to further elaborate on this topic. By using this online collaboration tool, each staff member will be able to contribute whenever he/she is available (a-synchronously, or synchronously). At the same time the playback function in Google Wave enables them to keep track of the lines of argument that have been discussed.

The tag the wave with the words: miRNA, genetic variability, Alzheimer disease, pathogenesis.

To support the discussion, wave members make use of robots providing access to databases presenting chemical (or protein) structures, DNA sequences and biomolecular structures.

Literature review

At a certain point in time the scientists feel that their effort reached a dead end. They decide to gain more information and to perform a literature review on the role of genes in Alzheimer disease. To prevent redundancy and to make sure everyone is timely informed about new facts, the review will be drafted in a wavelet. To support the management of references, robot “Igor” from Nature Publishing Group is added to the wave. Igor is configured to auto complete references based on the PubMed database. All supporting documents (PDFs, full text documents and slide presentations) are embedded in the wave.

Collaborators

At the same time it seems wise to consult colleagues from other research groups worldwide. The literature review already provided some names of leading scientists in the field of Alzheimer genetics, but to make sure that no authorities have been missed, the LinkedIn Robot “Linky” is added to the wave. The robot suggests names of scientists based on social distance and the tags that have been given to the wave earlier.

Based on the actions of step 3, a number of international leading scientists could be added to the wave. With this new collaboration and the information gained from the literature study, new insights occur. The scientists are encouraged by these new insights but need additional data to confirm their hypothesis. To acquire these data a project team is formed and Scientist R. is appointed as the primary investigator leading the team.

Proposal Writing

The drafting of a research proposal can be started now. Also drafting takes

place in a wavelet. All participants are invited to contribute and comment to the proposal. The drafting process can be tracked by using the playback function in Google Wave.

Project Management

The coordination of international projects is a complex activity with many dependencies. Therefore R. decides to use a project management tool (e.g. MsProject, Mindview or Project Kickstart...) to formalize the management of this project. R. creates workflows and assigns responsibilities to his colleagues. Tasks and alerts will be issued by the project management tool and pushed into the geneABC wave.

Now, they need funding to start data collection, storage of samples and sequencing of genetic material. Fundraising is a tedious activity and often, especially in international efforts, it is a major effort to obtain all the required information from participants and to complete the application for funding. Luckily, the Netherlands Organisation for Scientific Research NWO provides a gadget to facilitate the application of funding in Dutch and European research programs. The “NWOOKY” gadget is added to the wave. Now, all wave participants are able to fill in the required forms and for R. it is easy to monitor whether all partners fulfilled their tasks. The final proposal that has been drafted in step 4, can be copied to the NWOOKY gadget. For future reference, the proposal, which includes ownership, publication and IP policies is stored in the SURF repository. By adding the robot “SURFreposit”, users can store and retrieve documents to/from their Group folders in the safe SURF repository simply by adding the commands “store:” or “retrieve:” to Blips.

Data Collection

After a few months, the data and genetic material of 100 patients have been collected, checked, anonymized and cleaned. Raw and aggregated data can now be queried and analyzed. Besides patient centred data, log files from laboratory machines (simple XML logs) became available as well.

Data processing and analysis

The amount of data is immense and misinterpretation is a major threat. Therefore, researchers from different research-sites are invited to co-analyse these data in the wave. The tools for analysing these data in the wave are brought by the “SQLbot” robot that is able to query databases (e.g. Sql server, Progress, Oracle or Intersystems Caché databast). In addition a connection with the Blast database (Basic Local Alignment Search Tool) from the National Library of Medicine can be established from within a wave by adding the “Blastbot” to the wave. From there batch processing for analyses on the data (e.g. with statistical packages like SAS or R) can be initiated. The robots “SASbot” and “Rbot” are available to accomplish this task.

Drafting manuscripts

Preliminary results are given within a wave. The drafting of reports and manuscripts again takes place in a dedicated wavelet. The same applies to the drafting of graphics and figures with robot “Graphy”. All participants in the

wave can contribute, comment and track the process that leads to the final results.

Dissemination

As soon as all collaborators agree with the content of the final draft, the researcher decides to submit the manuscript to Nature. Nature Publishing Group (NPG) provides dedicated submission robots. The formatting of papers, sharing of live diagrams and charts as well as submission processes, both to journals and to other NPG repositories are supported by wave robots. Even the running of the peer review process for NPG journals is facilitated by Google Wave.

The drafting and publishing process is the area where the most obvious and rapid gains can be made. Google Wave was designed to remove the classic problem of sending around manuscript versions with multiple figure and data files by email so you would expect it to solve a number of these obvious problems.

6 CONCLUSIONS AND RECOMMENDATIONS

With Google Wave, Google introduces a very interesting new concept in the field of online collaboration. The fact that both human to human, human to robot, and possibly even robot to robot interaction can be handled in a similar way, all based on open standards and in real time, introduces some very exiting possibilities.

The Google Wave protocol suits nicely in SURFnet's Collaboration Infrastructure philosophy (link) on how services should interact and be represented. The specification leaves enough freedom to implement the authentication and authorisation schemes that would be preferred in the NREN environment, such as SURFfederatie. Serious consideration should be given to Level of Assurance for waves. Lack of this will seriously hamper trusted communications between wave servers and clients. Perhaps the access federation model commonly used within NRENs can serve as a template for solving this.

It is evident that in itself this protocol has large potential. However, the lack of implementations makes the real business value currently unclear. Should the wave protocol become implemented on a broad scale, a so called 'killer app', or better 'killer wave' is likely to arise rapidly. One of these may very well be the wave robots which are, when operating as 'agents', aiding a user in his daily workflow. This functionality may turn out to be a strong tool for both educational and research proposes. However, there are currently not many 'serious' robot implementations. As part of further research the implementation of a real world, not-for-fun example of a robot within the NREN context would help to better understand the wave potential.

By combining Google Wave with SURFnet services interesting opportunities may arise as has been showcased in this document. Again lack of implementations hampers determining true business value.

Finally, Google Wave is currently still in 'stormy weather' and specifications are still changing frequently. That said, an open source implementation of the wave server has been made available and SURFnet would do well in testing this, for example for usage in between campus domains.