Please double-click on the note callout for speaker notes!

Version 1 June 4 2009

ExperiaSphere and Google Wave

This is a public document and may be shared freely in its current for, and quoted if attributed to CIMI Corporation. Contact us for other uses.

Note: The material we include on Google Wave here is our interpretation of Wave based on Google's published material, as available at the time of our writing this document. We take no responsibility for the accuracy of this material and any who use this to understand Wave does so on their own initiative and without support of ExperiaSphere or CIMI Corporation. Trademarks are properties of their respective owners.

What We'll Cover Here

- An introduction into Google Wave
- A summary of how the basic concepts of Wave relate to the basic concepts of ExperiaSphere
- How ExperiaSphere might interwork with Wave
  - To implement Wave features
  - To incorporate Wave into experiences

Important Things to Keep in Mind

- Google Wave isn't in its final form at this point so conclusions on how it works are necessarily tentative
- There are two issues to consider with regard to ExperiaSphere and Wave
  - Wave and SocioPATH social frameworks
  - Wave and ExperiaSphere resource/service control frameworks
- There may be some relationships between Wave and **both** the above

What's in a Wave?

Obviously we believe Google's Wave is important, even revolutionary, but to understand why you have to look beyond the press coverage and explore the developer-level material to understand what Wave is—an **architecture for communications and collaboration**

Wave Essentials

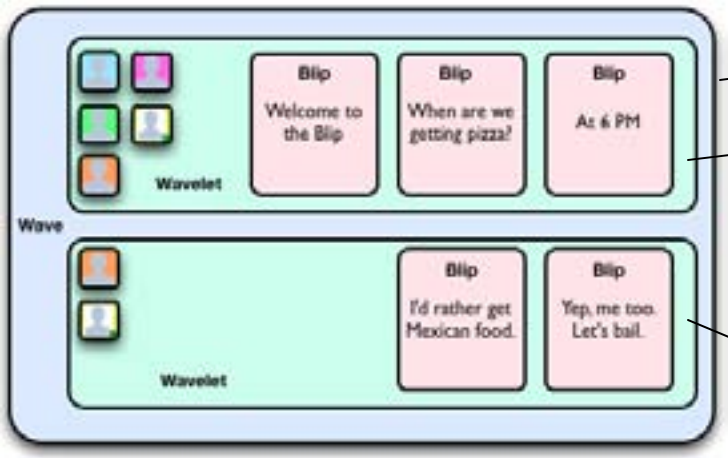
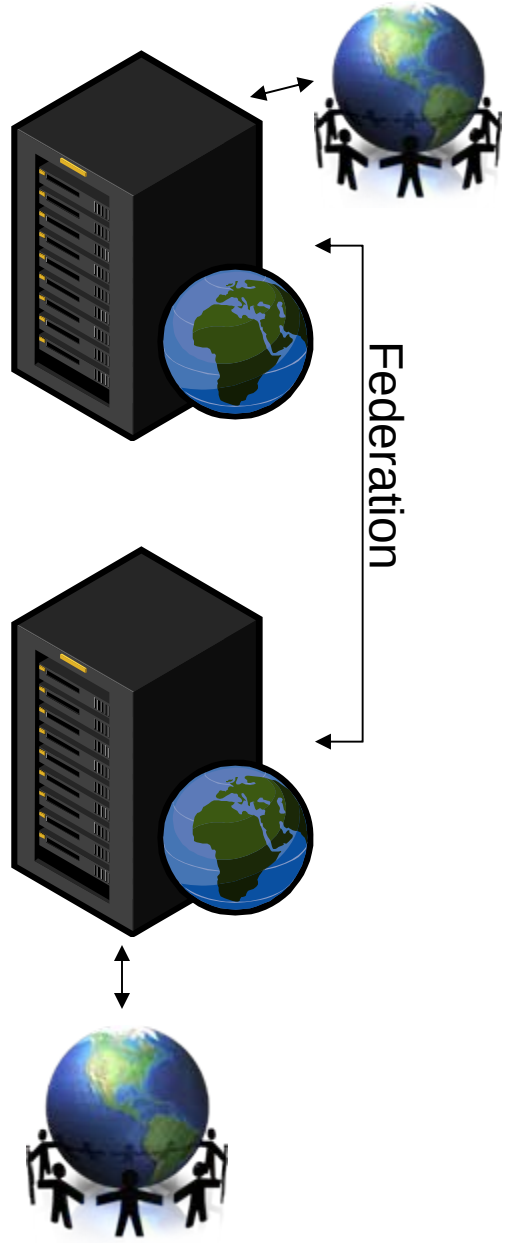


Figure Courtesy of Google



Waves are XML objects that are hosted on a server, and that act as “containers” to organize a group of conversational relationships among users who are members of hosted communities on multiple servers. Conversations are similarly structured and hosted, called wavelets—which can be federated from multiple hosts.

Wave GUI

Wavelets are the basic element of communication in Wave, and each Wavelet is hosted by the Wave server of the user who owns it. A copy is syndicated to the Wave Servers of other users who are participants, and the copies are synchronized through a Federation protocol, an extended XMPP

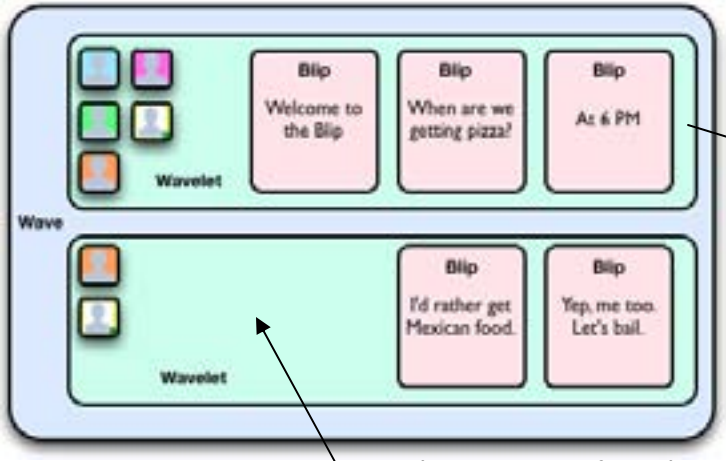
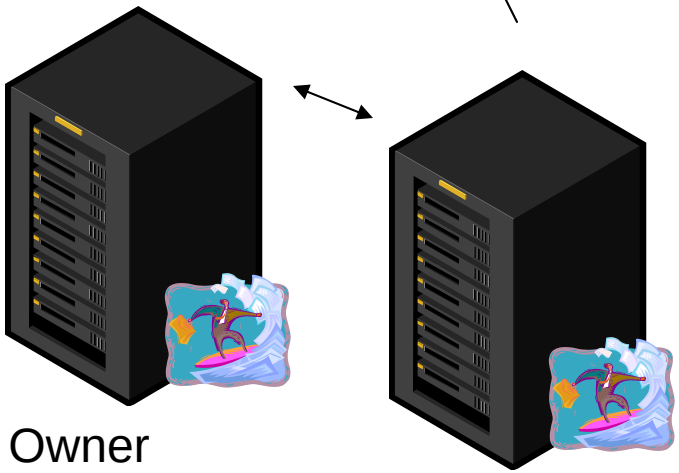
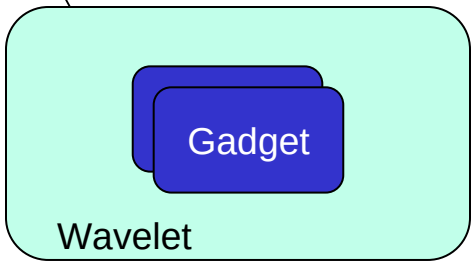


Figure Courtesy of Google

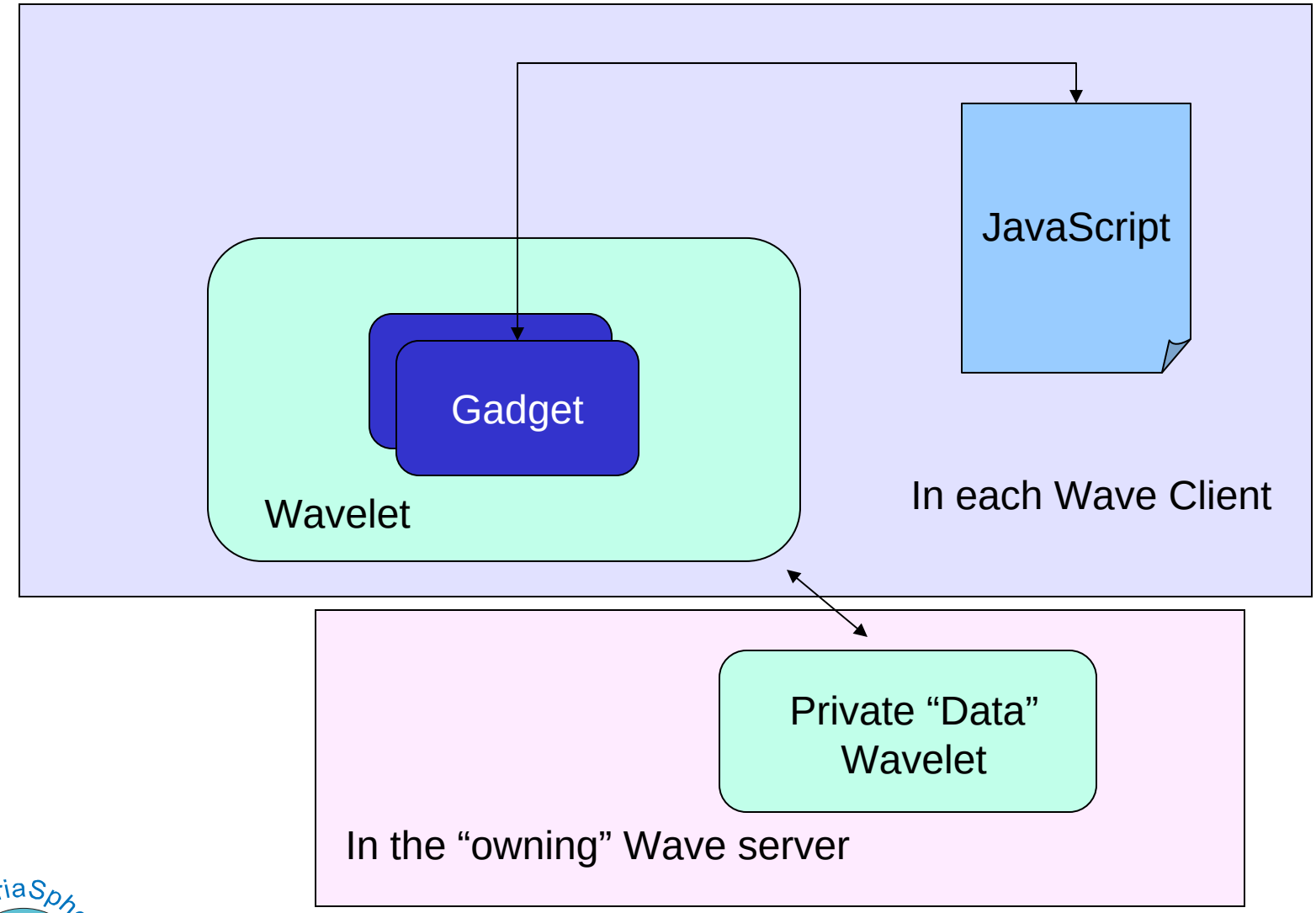


Owner

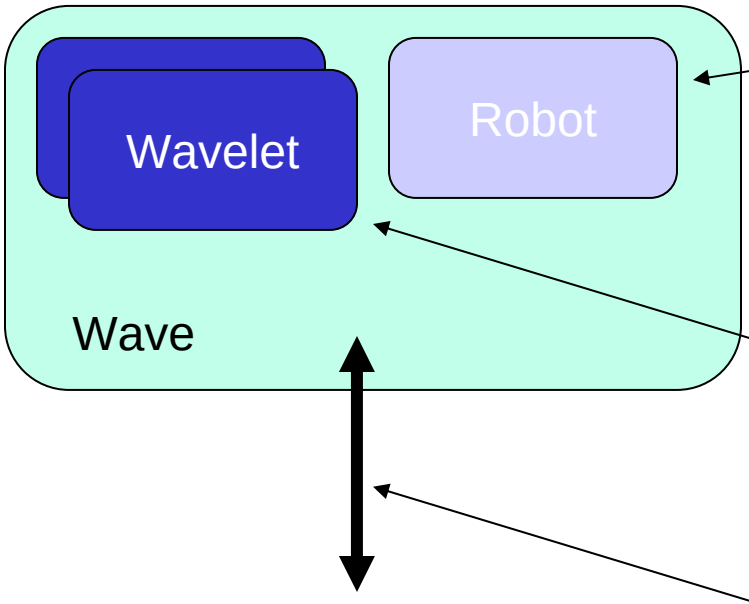
Member



Wave GUI In Detail



Waves and Processes (Robots, Gateways)



Robot processes and service gateways that provide management and linkage to other non-Wave environments

Multiple public and private wavelets, referenced in the Wave and hosted on the Wave servers of their owners

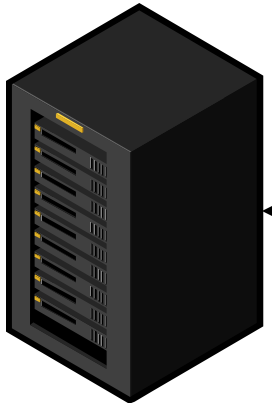
Wavelet updates are federated between the Wave servers of the participants and the Wave server where the wavelet is authoritatively hosted (its owner's server)

Gadget

Wave Client

A Gadget in a wavelet can create a signal for a communications service and populate the Wave data store with the needed information (address of user, for example)

Federation Link



ROBOT

A Robot running in a Wave server (or in theory in any compatible server, including and especially Google's App Engine) can then see the change in the wavelet and run the logic to make the connection

ExperiaSphere

Think Outside the Bit

Wave and ExperiaSphere Architectures Compared

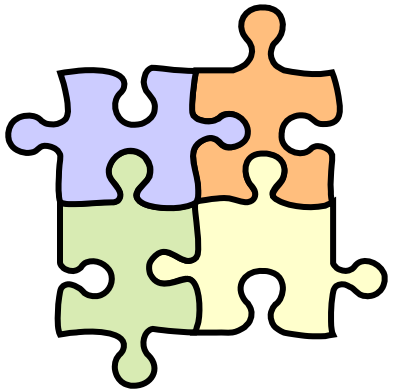
We didn't set out to anticipate Wave, nor we're sure did Google set out to replicate ExperiaSphere, so we have to look first at the basic approaches and how they differ

About the Next Slides

- Our goal is to look at how Wave compares with our ExperiaSphere work
- We'll present how Wave operates at a high level in three areas (service, hosting, and GUI)
- We'll then present how SocioPATH group-social communications works, and how ExperiaSphere resource/service control works in those same areas

The “Service”

In Wave…

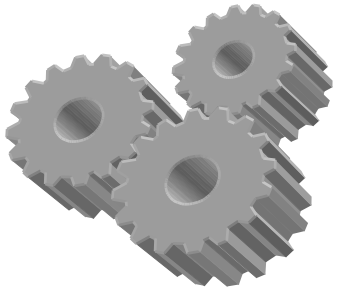


*A mosaic of conversations inside “wavelet” containers, collected into a **context-object** called a Wave*

In SocioPATH…

A set of collections of Entities called “Groups”, defining a social context for all relationships between the members

In ExperiaSphere



An agreement to commit resources to secure a specific user experience

The Hosting

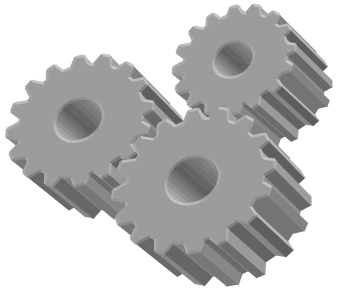
In Wave…

Users are grouped into what we'd call "Seas" around Wave Servers, and the users' own server hosts at least a copy of the Waves/wavelets they are using

In SocioPATH…

User groups are hosted by Group Entity Brokers, and individual users by Local Entity Brokers. A GUIBroker provides access to the physical client devices

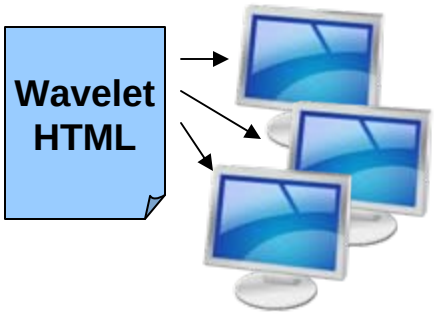
In ExperiaSphere



Experiams are hosted by the administration that creates one; a Proxy is used to represent a federated functionality from a different administration

The User's Interface

In Wave…



A Wave Client provides the basic display, and a single wavelet-based structure provides a template for the GUI to each participant, with the Wave server for the participant hosting the local copy

In SocioPATH…

A GUIbroker provides the "panel" that represents the relationships to each user/device independently, and the panel can work inside a browser or any other display tool

In ExperiaSphere

Service ordering would normally be done through a web server linking to an OrderBroker; the same structure could be used for other service GUIs but an application GUI isn't normally part of ExperiaSphere services

Wave Concept	ExperiaSphere Concept	Harmony
Context-Object “Wave” is the focus of presentation and implementation	In SocioPATH, the user constructs a panel that presents a user’s “social universe” and services are then mapped into it	The panel/user association is arbitrary; it would be relatively easy to map a unique panel for each experience; harmonizing the user of the two different approaches might require thought
Waves are hosted on a Wave Server, and made up of wavelets	GUI Panels are hosted on the GUIBroker, presence and state on The Local Broker and the user’s specific templates are hosted on the Owning Group server	The hosted notion is essentially the same; the GUIBroker in SocioPATH provides the display management for the user and the Local Broker the presence hosting
Waves support collaborative editing and concurrency control for documents	ExperiaSphere doesn’t contain a document editing application; there is a basic concept of “concurrency control” associated with Proxy/Host	If concurrent editing is a goal, it would be logical to link it to ExperiaSphere from Wave or another platform rather than duplicate it in ExperiaSphere
Wave hosts federate the coordination of Waves and wavelets	ExperiaSphere federates roles in the experience control portion and federates user profiles among Group Entities in SocioPATH	ExperiaSphere could implement the XMPP and extension approach taken by Wave

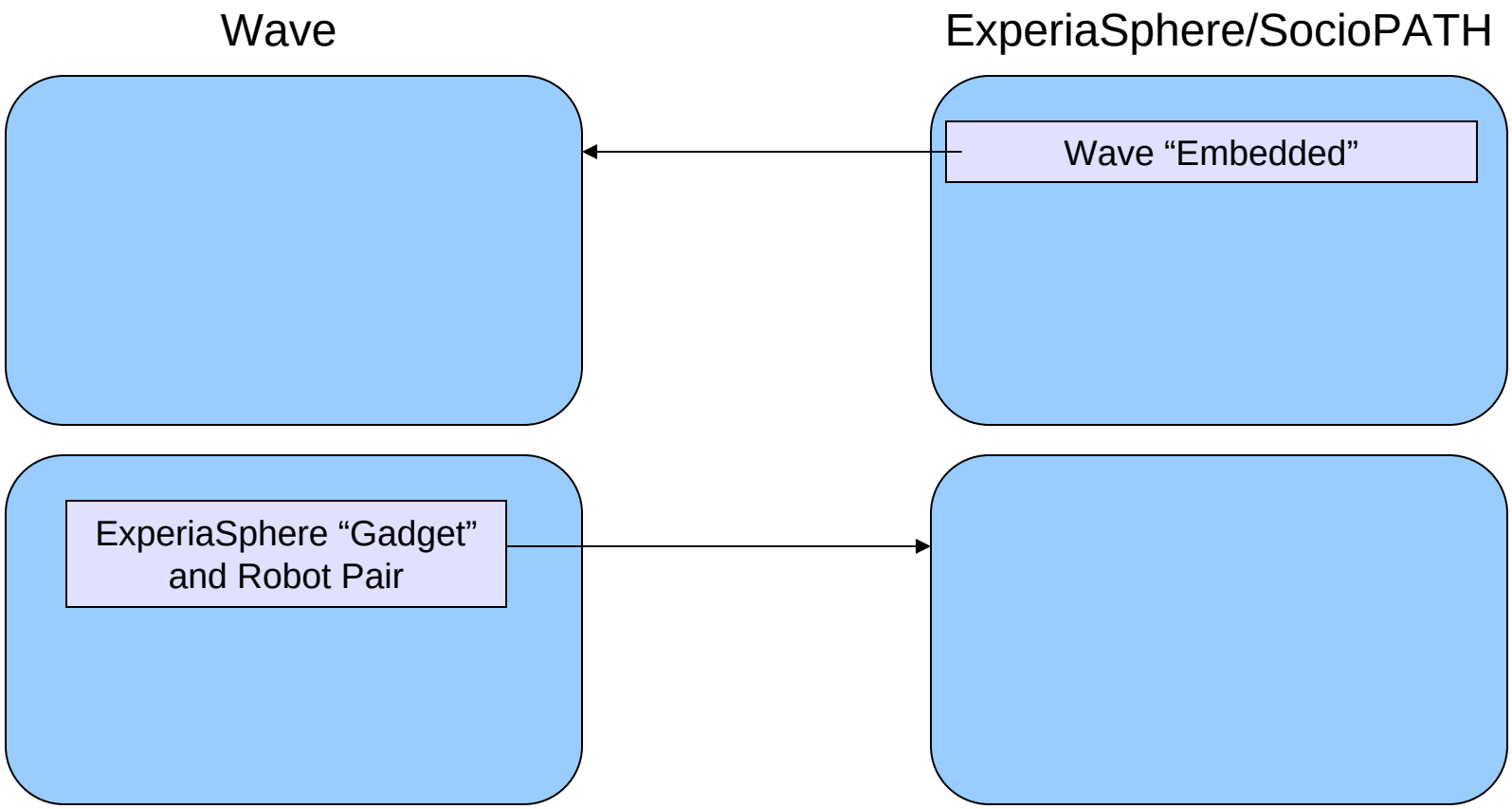
Some Strategies for an ExperiaSphere/Wave Partnership

Just because the frameworks are not identical doesn't mean that there aren't ways that ExperiaSphere and Wave can't behave as partners…no matter who wants to lead!

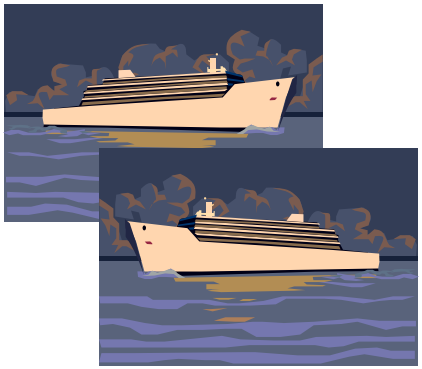
Two Options

- An “outside-in” approach that harmonizes Wave and ExperiaSphere from the perspective of the user
- An “inside-out” approach that harmonizes Wave and ExperiaSphere at a technical level
- **Remember that we’re looking at an implementation recommendation here, so the inside/outside difference is where we start and not where we end up!**
- We’ll look at both and compare the results!

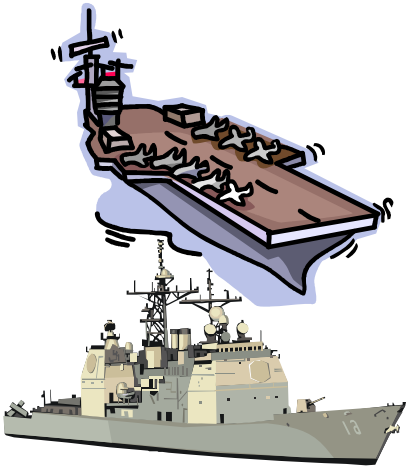
Outside-In GUI Harmony



Outside-In to the Next Level

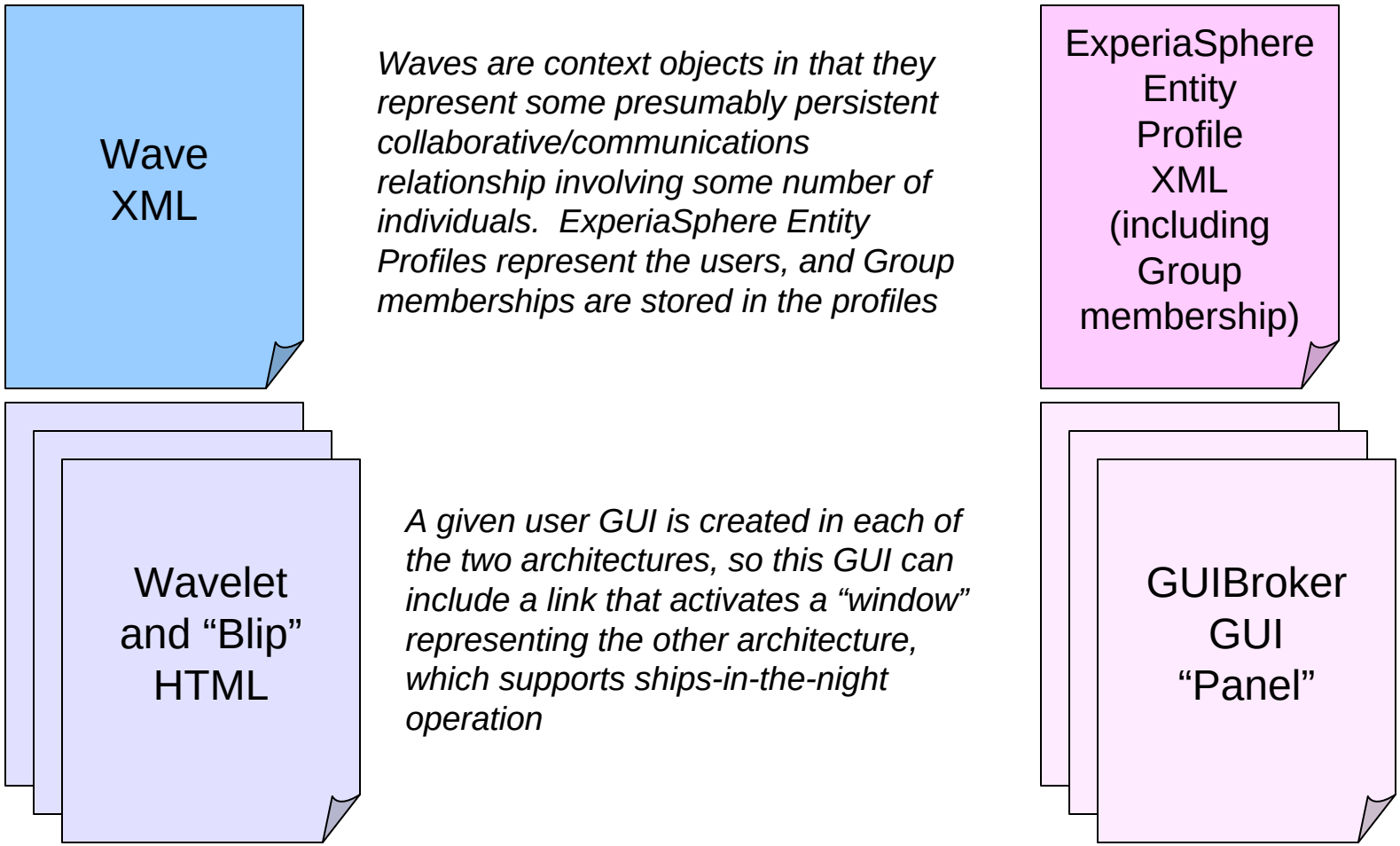


Ships in the Night: Wave and ExperiaSphere could be run as two different and parallel applications, with a link (embed or Gadget) in one simply invoking the other. **Nothing needs to be done to either to make this work other than the cosmetic embedding of links!**

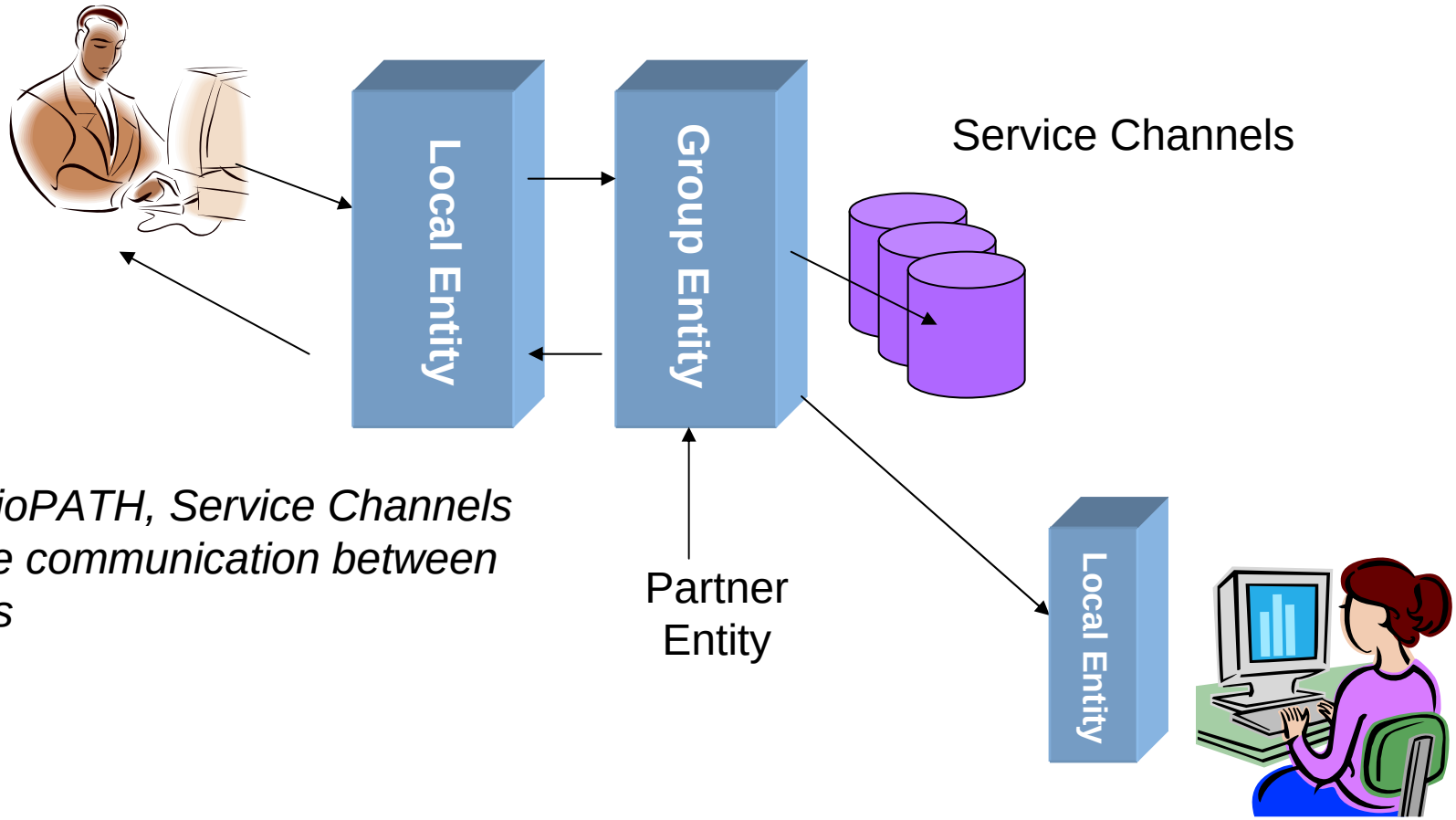


Partner Processes: Wave and ExperiaSphere could be run in a symbiotic way where there was functional integration between the two. **To make this work we must define how one process would look to the other**. That leads to our "inside-out" view.

Creating a “From the User In” Harmony as Ships in the Night

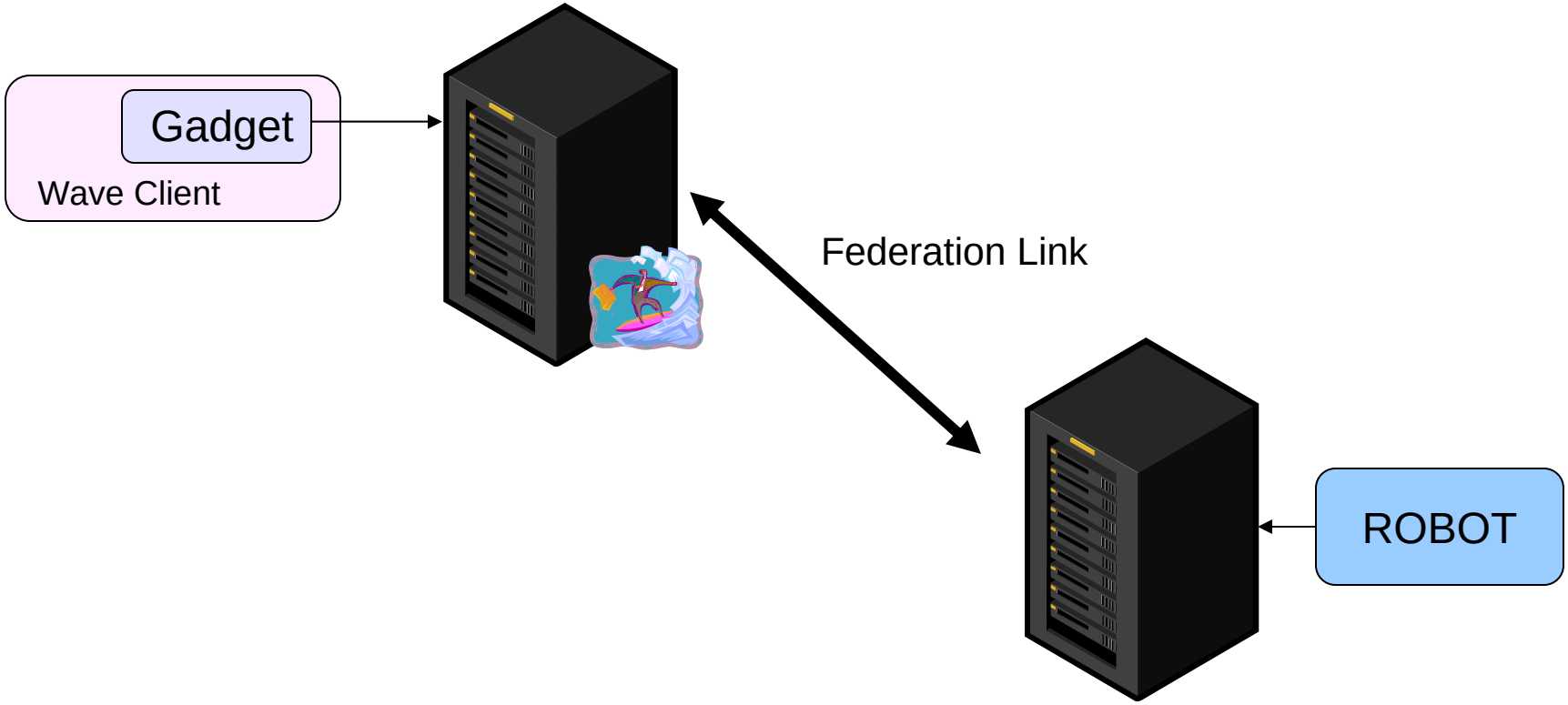


Inside-Out Integration: Service Channels



In SocioPATH, Service Channels provide communication between Entities

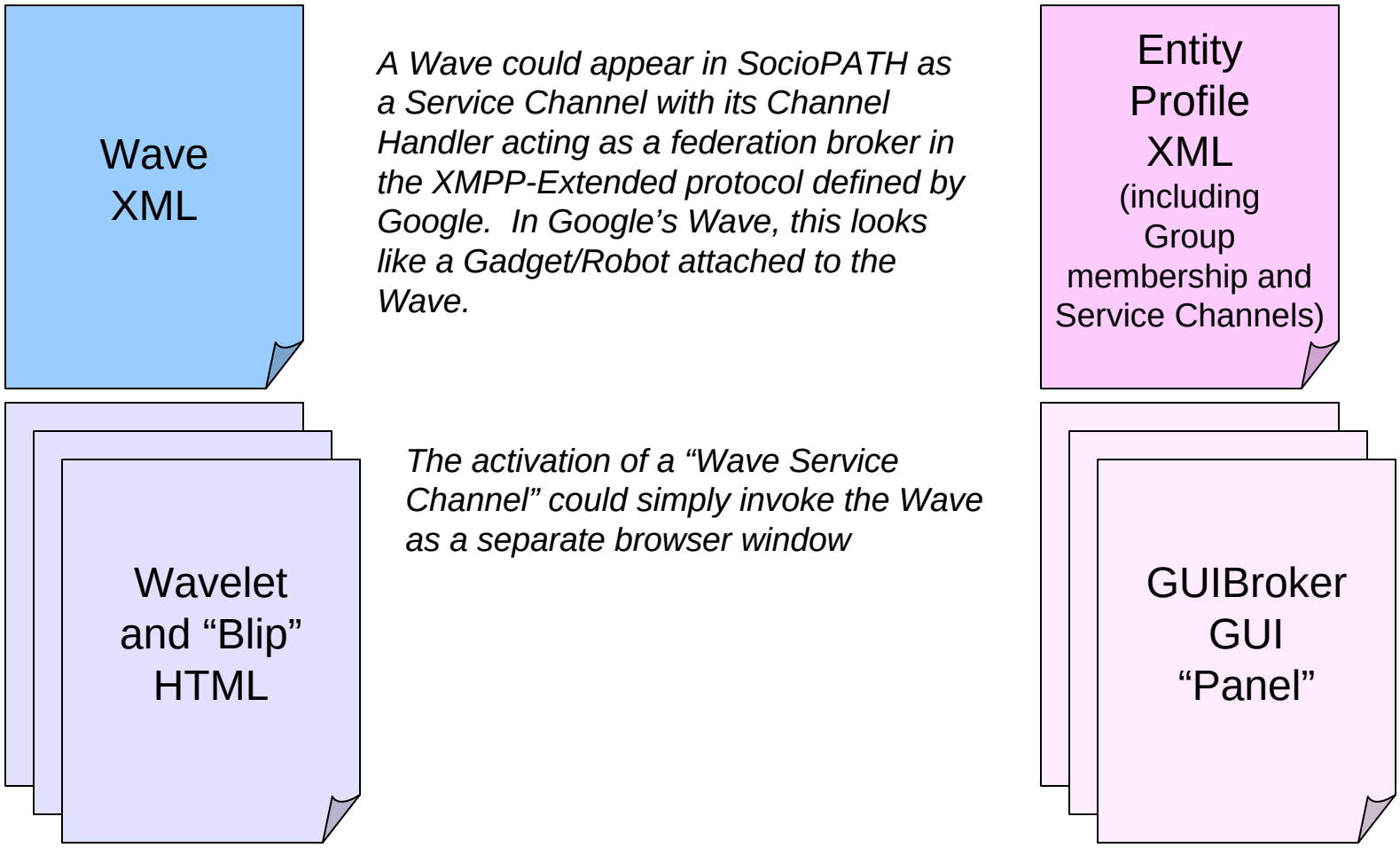
Wave Gadgets and Robots can Link to “Communications Services”



The Service Channel as a Unification Strategy

- Service Channels represent “connection services” in ExperiaSphere/SocioPATH
- Google Wave can integrate connection services (like their Twitter example) through Gadgets and Robots
- Thus, Service Channels can be viewed as a language spoken by both architectures, and a logical way to create harmony

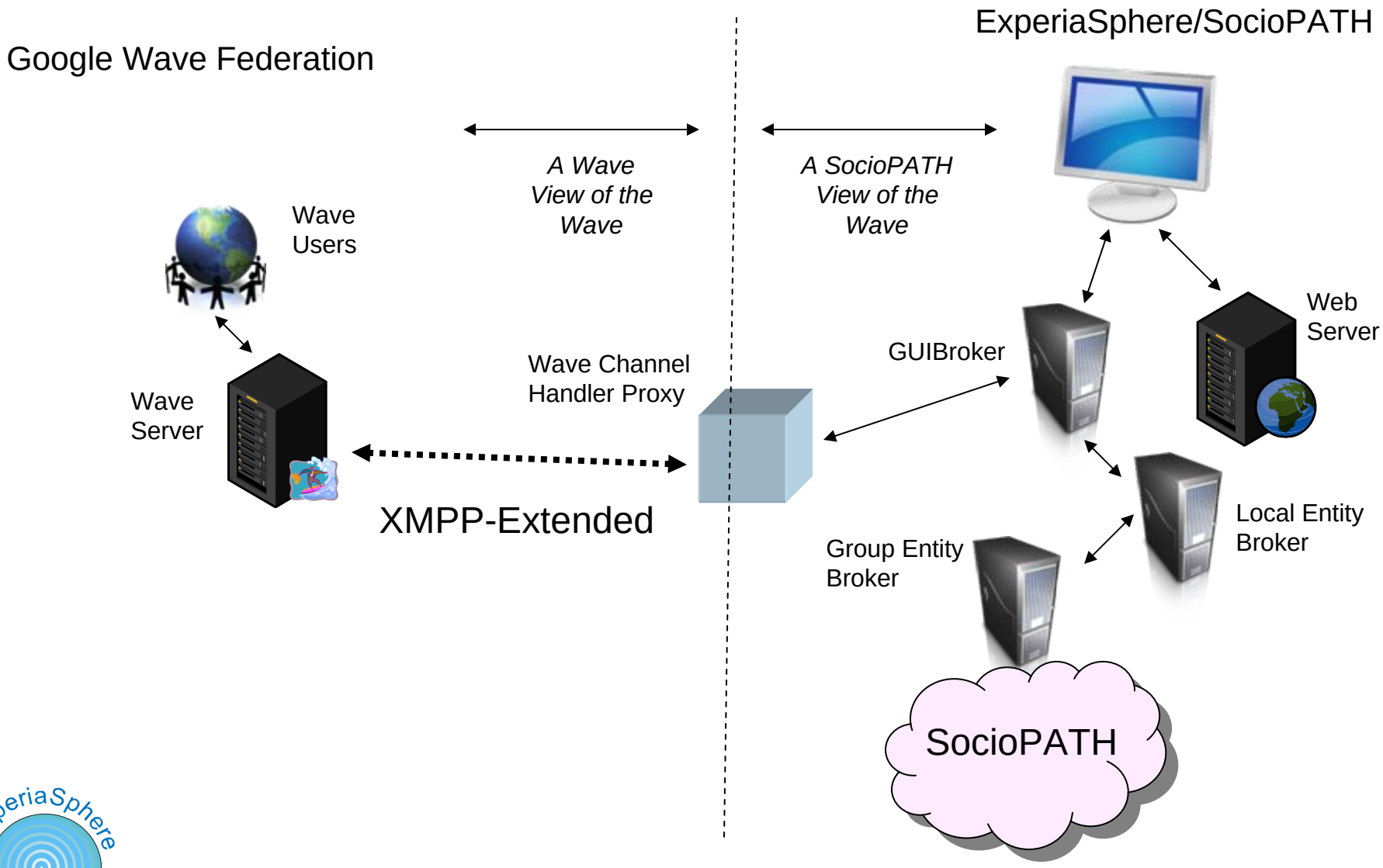
Inside-Out Harmony: The “Service Channel” Approach



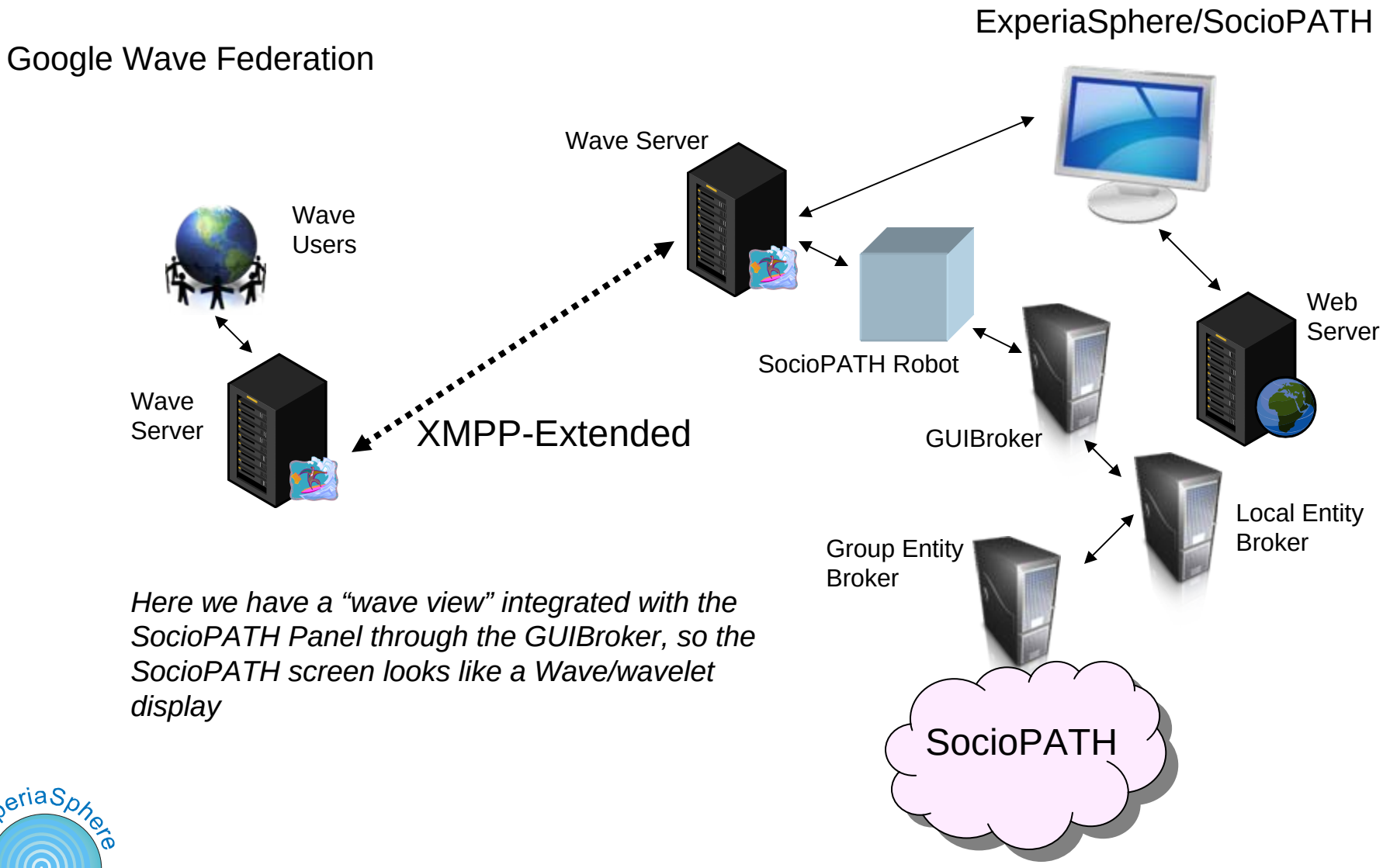
Can We Tighten Integration?

- A common Service Channel gives us functional integration at the connection service level
- The two GUIs are not integrated so the Wave user doesn't know about SocioPATH; each just uses a common channel
- Integrating the GUIs in some way means making ExperiaSphere speak Wave's Federation protocol language

Service Channel Handler as a Proxy



Using an “ExperiaSphere Wave Server” for Integration



Here we have a “wave view” integrated with the SocioPATH Panel through the GUIBroker, so the SocioPATH screen looks like a Wave/wavelet display

Summary

- Google Wave is still in a preliminary state, lacking some of the tools promised for integration
- Simple integration by cross-connecting GUIs is possible today
- More complex integration regarding shared Service Channels and shared wavelets is the preferred solution
- When the Google information has advanced to the point where the APIs and process descriptions are published in fairly stable form, we propose to proceed on the basis of the complex model of integration outlined in the “Using an ExperiaSphere Wave Server for Integration” slide