

Waking Up the Web

The Design of An Automated, Activated World Wide Web

Scenera Research LLC

Cary, North Carolina

www.sceneraresearch.com

We have developed an automated system that allows users to access, browse, and conduct business transactions on the Web in real time. It communicates with network resources such as Web pages, files, email, and digital peripherals, and keeps those resources current by automatically updating information from them without the need for user requests. It is asynchronous and can support two-way communication such as instant messaging and remote procedure call.

Unlike the current Web, which is reactive and requires information and services to be located and requested, it is active and provides real time information and services in response to change. It operates from a single protocol model that can serve to consolidate a variety of existing protocols just as HTTP, which also has been a consolidating protocol. Like HTTP, its protocol is simple, operating with three text-based commands. It extends today's Web by adding an asynchronous, loosely-coupled communication system that allows the Web to be active.

Introduction

The World Wide Web as we currently know it is a passive, reactive system. Its information and services must be requested by users, and users must know or determine where information and services are in order to access them. Consequently, the Web is heavily dependent on indexing, search tools, and the well-honed search skills of its users.

As most Web users know, keeping up with information and services on the ever-growing and evolving Web is an enormous job. Most of us have the time and skills to access only a handful of all that the Web has to offer. Some users with plenty of time on their hands and well-developed search skills can take advantage of more of what is available, but even experts cannot tap into everything they may want or need, or keep the things they do find up to date. If the Web was more automated, users could accomplish much more in less time. What they need is an active Web: one that can work for them. The potential is there, and we have found a way to harness it using a simple, unique publish/subscribe protocol with the well-known service, presence. Although presence services operating over publish/subscribe protocols are in wide use, their full potentials have not yet been applied.

Unforeseen potential is not uncommon with the frenzied growth of the Web. For example, hypertext transfer protocol (HTTP) was initially developed to navigate between hyperlinked documents. Thanks to its simplicity and flexibility and through the consolidation of many protocols, HTTP ultimately evolved to become a backbone of the entire, ever-growing World Wide Web.

Like HTTP, publish/subscribe and presence are simple, flexible networking tools. Both are used in high volume, and both have the substance to do much more than they are typically asked. With publish/subscribe as the backbone and presence as its core service, together they can become an automated, activated Web.

Asynchronous Communications

Today the Web operates with synchronous communication using HTTP, a request-response protocol. Consequently, its information and service providers must wait for specific requests before they can respond. But publish/subscribe is asynchronous at its core, and thus can provide presence with information as soon as that information changes on the Web.

Synchronous communication works well in support of certain browser tasks, such as sending a request to a server for a Web page and waiting for a reply from the server to display the page. But other browsing tasks are supported better using asynchrony. For instance, during an online purchase, original and updated information about a seller's item are needed frequently by the buyer. Synchronous protocols often delay purchases because they require sellers and buyers to request information updates manually. Asynchronous protocols can automatically provide the information they need in real time.

Asynchronous communication offers numerous possibilities for the Web, well beyond what we ask our browsers to do today.

The Activated Web

Because HTTP is a passive, reactive protocol, when using it to find something on the Web a user must know or determine where it is and then request it. The asynchronous system we call the Activated Web can detect information of interest to users and provide them with what they want in real time and with continuous automated updates. To do this, it uses presence services with a publish/subscribe protocol.

Today, presence services are primarily used to provide current status information to users. For example, they provide users of instant messaging (IM) services with the status of other IM users, indicating whether each user is on line, off line, busy, away, away for a minute, doesn't want to be disturbed, and so on.

Many different protocols provide presence services status information on IM. Some of the most common presence-compatible protocols are publish/subscribe; however, many of them are proprietary and vendor specific, and therefore are not compatible with one another. Publish/subscribe protocols for providing presence have been standardized by the Instant Messaging and Presence Protocol (IMPP) Working Group (1, 2), and today there are two major competing standards in use (3). For the Activated Web, we incorporated the IMPP's presence standards into one simple, general purpose publish/subscribe protocol that can handle all of its operations. At the same time, it can be adapted to provide a request-response protocol with capabilities that extend well beyond those provided by HTTP.

The Publish/Subscribe Protocol

Although IM applications that use presence services to determine a user's status and address predominately use publish/subscribe protocols to provide their services, those protocols are not used to deliver instant messages or any information or services other than status notification-based ones. Publish/subscribe protocols are expandable and used to accomplish much more than that in other arenas, and we envision using a publish/subscribe protocol to consolidate many of the application-specific protocols that are in use. Just as the consolidating impact of HTTP has been fundamental to the growth and expansion of the Web, we believe the impact of a consolidating publish/subscribe protocol can be just as significant for the development and expansion of an Activated Web.

Protocol Commands

Adopting a general purpose protocol has proven to be important because the success of the Web is largely based on the simplicity and adaptability of HTTP. HTTP has five commands, and only two are required to run the Web: GET and POST. The publish/subscribe protocol has nine commands, and only three are required to run the Web: PUBLISH, SUBSCRIBE, and NOTIFY (Table 1). SUBSCRIBE is a GET command, and PUBLISH and NOTIFY are both POST commands. With these commands the publish/subscribe protocol is simple and flexible, like HTTP. Flexibility allows these protocols to be consolidated, which is particularly important for devices that support a limited amount of information.

The three required commands of the publish/subscribe protocol are described in Table 1, and the six optional commands are described in IETF RFCs 2778 and 2779 (1, 2).

Table 1. The three required commands of the publish/subscribe protocol.

Command	Definition
PUBLISH	Allows a presence entity to provide or update presence information, such as status or contact information, to a presence server.
NOTIFY	Allows a presence server to provide information from a presence tuple to a watcher. Notification may be point-to-point or broadcast.
SUBSCRIBE	Allows a watcher to subscribe or unsubscribe to notifications regarding specific presence information.

Because these commands can be run on any request-response devices that are configured to exchange information with a presence server, and because the publish/subscribe protocol requires so little memory, the Activated Web can run on a wide range of devices, including laptops, cell phones, PDAs, and digital cameras, to name just a few.

Clients and Data

Presence has two types of clients: the presentity and the watcher. The presentity provides the information to be stored and distributed through the presence server. The information it handles can

include any information needed by the server, such as the status of a user, the means of communication, and the user's contact address.

The server can store the information any way it wants, and in keeping with our goal of simplicity, the Activated Web stores it in "tuples." A tuple is a data object with two or more components, such as a user ID, a user's status, a contact address, a subscription, or any other data used by presence services. The size of a tuple is limitless: a Web address or an entire encyclopedia can be a tuple. More importantly, whenever a subscription is updated on the Activated Web, the only item that needs to change is its tuple.

Watchers receive information from the presence server. There are two types of watchers: subscribers and fetchers. Subscribers request notification of changes in presentity information from the service that establishes the subscription, and when there is a change in the tuple, the update is pushed to the subscriber. Fetchers request and pull current presentity information from the presence server. A special type of fetcher that is particularly important to the Activated Web is the poller, which fetches information on a regular, or polling, basis and helps keeps the system active.

The presence server also manages, stores, and distributes watcher information and activities, such as fetching and subscribing, to other clients using the service.

Users

The users of the Activated Web can be people, software programs, mobile devices, or any other resource that can communicate with a presence server. This allows vast possibilities for future Web applications.

Configuration

The basic configuration of the Activated Web is shown in Figure 1, and it looks very much like a typical networked system. As mentioned earlier, it has two significant benefits over today's networks: 1) it is asynchronous, and therefore operates in near real time and requires no user input for sending and receiving, and 2) it requires just one protocol, a publish/subscribe protocol, for its operation.

When setting up a subscription on the Activated Web, a client sends a request over a presence service network to engage in activities with receiving clients who have indicated an interest. The request gives a description of the activities, which can be connected to instant messaging, cell phones, cameras, PDAs, Web sites, e-mail, documents, libraries, and any Web-based resource. Once the request is received and accepted, the tuple is updated to include the new client.

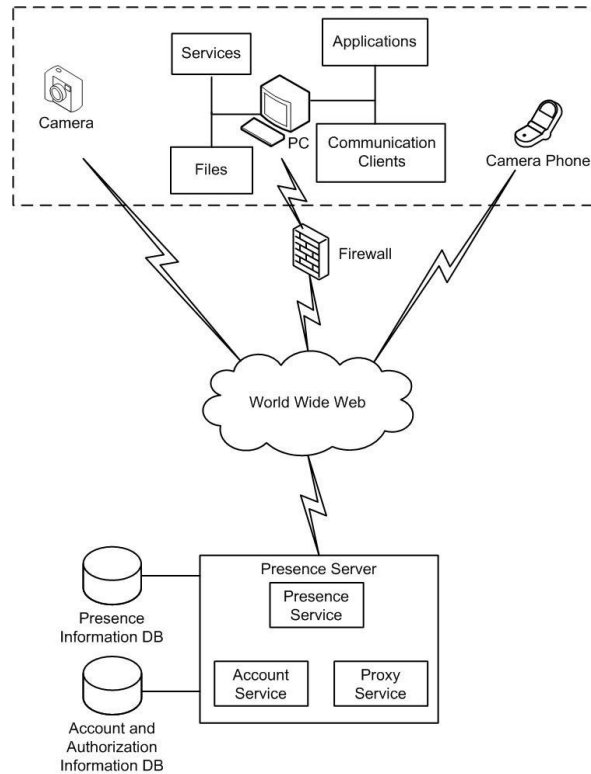


Figure 1. The general configuration of the Activated Web with its presence service and varied digital peripherals.

Operation and Applications

The general purpose publish/subscribe protocol can help improve many common Web operations and applications. It can help make browsing safer and extend the capabilities of existing protocols through tuples and message flow. Examples of what presence and publish/subscribe can contribute to Activated Web operation and applications outlined here illustrate the breadth of the system's immediate capabilities.

Web Browsing

This system provides browsing through the publish/subscribe protocol. It has a graphical user interface (GUI) that is configured to receive the ID of a tuple associated with a Web page or other network resource and a protocol agent that is coupled to the GUI. The protocol is configured to use the ID to request a subscription to the tuple and is configured to receive information related to the network resource and its link. A communications protocol stack coupled to the protocol agent is configured to request the subscription to the tuple and to receive the information and make links using the asynchronous publish/subscribe protocol.

To view a Web page or document in real time while browsing, initially the user sets the browser to control the pace at which the page is updated. Without pacing, the browser may be updated more quickly than the user can view or process the information it provides. With today's browsers, this can

cause the buffer to overflow and submit a request to stop sending any updates at all, which is a limit that cannot be controlled by users.

Today, HTTP is used to pace the rate of information received on a device connected to browsers. However, pacing synchronous protocols does not allow users to receive real-time data because separate, user-initiated requests must be made for each update and each page or file that the user is connected to.

In the Activated Web system, pacing can be controlled in different ways. It can be controlled by the user through the browser, it can be associated with a subscription, and it can be browser-specific. Published materials displayed through a browser can also be queued according to their pace settings by delaying their presentation based on pace.

Simplified Instant Message Exchange

The single publish/subscribe protocol of the Activated Web makes it easier to exchange instant messages. Currently, applications that provide instant messaging and presence require two protocol agents at each client device, at least two servers at each service point that provides the services, and two protocol command sets to deliver both the IM and presence services. This replication of infrastructure can make it difficult or impractical to integrate the storage of IM and presence data. Also, supporting two separate protocols requires more memory and makes it more difficult to secure the devices with the network that their commands flow on, and managing message traffic becomes more challenging. Therefore, exchanging messages using a single protocol is a clear advantage.

Automated Business Transactions

HTTP limits the freedom of buyers and sellers in terms of fees they pay, where they can shop and sell, and the methods they can use to advertise and locate their goods and services. HTTP browsers also make it overly laborious to obtain real-time information on items that users bid on, are considering to buy, or have purchased or won in an auction.

Users of the Activated Web will not need to reopen or refresh their Web pages to see updated information. Fresh information is provided automatically and in real time. This is valuable in many scenarios, most notably in monitoring dollar fluctuations for online businesses, such as stock trading and goods auctioning where prices can change rapidly and users often miss updated information.

To keep the Web page automatically updated, the requestor (a buyer or device) sends an initial request to a client to take part in a sale or auction, then receives a response. All information about the sale or auction, plus the means for interfacing, subscribing, and presentation, are provided by the client and both the client and the requestor have the protocol needed for communication. The protocol agent and server are configured to receive a subscription to publish the appropriate sales and auction information and to provide an interface across the network. Figure 2 is an illustration of how an automated business transaction network can be configured and used for shopping and auctioning online.

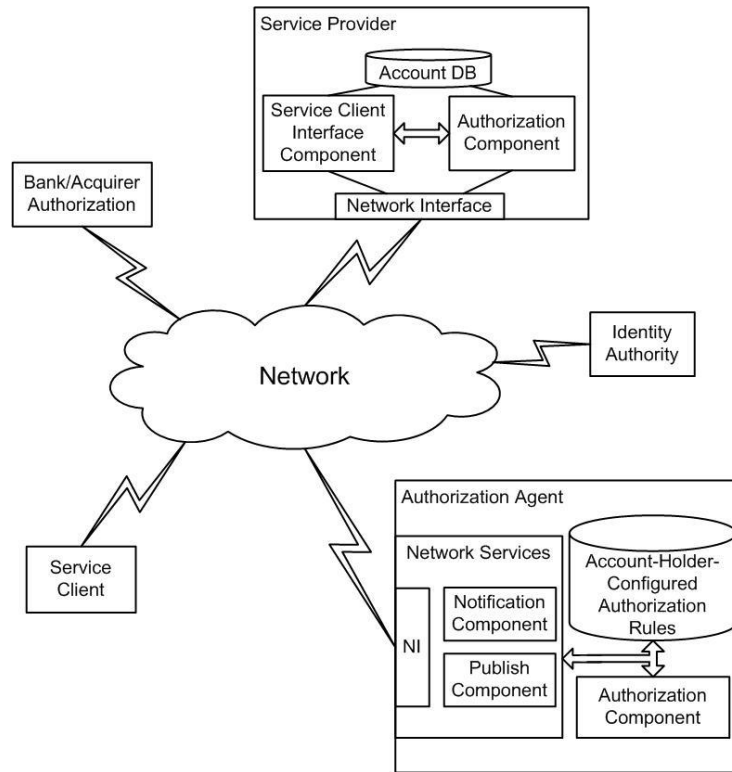


Figure 2. The general configuration of an Activated Web system for automated online business transactions.

Business transactions performed on this system may also include the online context-sensitive instructions needed to perform purchases and take part in auctions, such as how to receive information about sales and auctions and how to participate in them.

Information about the business transaction is managed by the publish/subscribe protocol. When a subscription to an auction is authorized, all subscribers to the auction are notified, including the seller, and all bids are updated automatically so that all of the buyers and the seller always have current information. When new information about a sale or auction is received from a subscriber (a.k.a. “buyer”), it is updated by the protocol which includes a link that represents a tuple associated with the sale or auction.

When an item is purchased, it moves to the part of the system that places the order, purchases the item, and processes its shipment. Auctions and the transactions surrounding them are described in detail in a forthcoming paper (4).

Beyond Business Transactions

Automated bidding, purchasing, and shipment on the Activated Web shows the vast potential of the system. It is easy to see how it can be applied to many large organizations and processes, such as networked library systems, stock monitoring and purchasing operations, real estate searching and buying, and travel planning and scheduling, to name a few.

The simple tools needed to develop this system exist today. All that is needed is to bring them together and put them to work.

Future Development

Security

We recognize the need for development of security across the Activated Web system. A Web-wide publish/subscribe service manages the dissemination of data across heterogeneous platforms, distinct domains, and numerous publishers and subscribers, all of which are dynamic. Users, accounts, subscriptions, and services all must be protected, which is no small task.

Other Areas

Other areas we believe publish/subscribe services have significant potential include privacy services, work flow management, load balancing, home automation, and supply chain management. We are actively exploring these areas.

Acknowledgments

The author would like to thank Julie Tomlinson for her writing, editing, and research.

References

1. M. Day, J. Rosenberg, and H. Sugano. 2000a. A model for presence and instant messaging. Internet Engineering Task Force, RFC 2778: <http://www.ietf.org/rfc/rfc2778.txt>.
2. M. Day, S. Aggarwal, G. Mohr, and J. Vincent. 2000c. Instant messaging/presence protocol requirements. Internet Engineering Task Force, RFC 2779: <http://www.ietf.org/rfc/rfc2779.txt>.
3. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler. 2002. SIP: Session Initiation Protocol. Internet Engineering Task Force, RFC 3261: <http://www.ietf.org/rfc/rfc3261.txt>.
4. "[Activated Auctioning Using a Web-based, Automated Request-Response System](#)." Scenera Research LLC, 2008.

Contact

Contact Scenera Research at info@sceneraresearch.com.