

Sussex Research

Supporting public availability and accessibility with Elvin: experiences and reflections.

Geraldine Ann Fitzpatrick, Simon M. Kaplan, Tim Mansfield, David Arnold, Bill Segall

Publication date

01-01-2002

Licence

This work is made available under the **Copyright not evaluated** licence and should only be used in accordance with that licence. For more information on the specific terms, consult the repository record for this item.

Citation for this work (American Psychological Association 7th edition)

Fitzpatrick, G. A., Kaplan, S. M., Mansfield, T., Arnold, D., & Segall, B. (2002). *Supporting public availability and accessibility with Elvin: experiences and reflections*. (Version 1). University of Sussex.
<https://hdl.handle.net/10779/uos.23312117.v1>

Published in

Computer Supported Cooperative Work

Link to external publisher version

<https://doi.org/10.1023/A:1021226206564>

Copyright and reuse:

This work was downloaded from Sussex Research Open (SRO). This document is made available in line with publisher policy and may differ from the published version. Please cite the published version where possible. Copyright and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners unless otherwise stated. For more information on this work, SRO or to report an issue, you can contact the repository administrators at sro@sussex.ac.uk. Discover more of the University's research at <https://sussex.figshare.com/>

Supporting Public Availability and Accessibility with Elvin: Experiences and Reflections

Geraldine Fitzpatrick[†], Simon Kaplan^{†‡}, Tim Mansfield[†], David Arnold[†] and Bill Segall[†]

[†]*CRC for Enterprise Distributed Systems Technology, The University of Queensland, Australia 4072. Ph: +61 – 7 – 33654310. Fax: +61 – 7 – 33654311.*

[‡]*School of Computer Science and Electrical Engineering, The University of Queensland, Australia 4072. Ph: +61 – 7 – 33652097*

{g.fitzpatrick,s.kaplan,t.mansfield,arnold,bill}@dstc.edu.au}

Abstract.

We provide a retrospective account of how a generic event notification service called Elvin and a suite of simple client applications: CoffeeBiff, Tickertape and Tickerchat, came to be used within our organisation to support awareness and interaction. After overviewing Elvin and its clients, we outline various experiences from data collated across two studies where Elvin and its clients have been used to augment the workaday world to support interaction, to make digital actions visible, to make physical actions available beyond the location of action, and to support content and socially based information filtering. We suggest there are both functional and technical reasons for why Elvin works for enabling awareness and interaction. Functionally, it provides a way to produce, gather and redistribute information from everyday activities (via Elvin) and to give that information a perceptible form (via the various clients) that can be publicly available and accessible as a resource for awareness. The integration of lightweight chat facilities with these information sources enables awareness to easily flow into interaction, starting to re-connect bodies to actions, and starting to approximate the easy flow of interaction that happens when we are co-located. Technically, the conceptual simplicity of the Elvin notification, the wide availability of its APIs, and the generic functionality of its clients, especially Tickertape, have made the use of the service appealing to developers and users for a wide range of uses.

Keywords: awareness, event notification service, Elvin, tickertape, chat tools

1. Introduction

The importance of supporting awareness for promoting a sense of shared place and work is widely acknowledged within the CSCW community. While the approaches to thinking about how we can support awareness when people are distributed are many and varied, there have been two main strands of work. One approach has been the use of media spaces as open video/audio channels across physical sites to provide for more ambient peripheral awareness. Another approach, most useful for focussed collaborative activities, has been the creation of purpose-built collaborative environments with embedded awareness support (?; ?; ?).



© 2007 Kluwer Academic Publishers. Printed in the Netherlands.

Event-based notification services are increasingly being used to provide the awareness support for these collaborative environments (??; ??; ??).

We also use event-based notification. Our focus here though is not so much on dedicated collaboration systems but on how we might go about making the workaday world (?) more ‘inherently’ collaborative when much of that workaday world happens within a distributed digital and physical environment, as it does in our organisation, the DSTC¹.

Specifically, this paper concerns the design of, and experiences with, a generic asynchronous notification service called *Elvin* (??; ?). Elvin was primarily designed as middleware for distributed systems. The later development of a graphical scrolling Tickertape client made the technical simplicity and general applicability of Elvin both evident and user-accessible. Very quickly, it began to be adopted and adapted by different research and prototyping groups within our organisation for application level purposes. Other clients such as Tickerchat and CoffeeBiff were also developed. These user-level applications included interactive chat, social and content-based information filtering, giving computer-based actions perceptible form, and making physical actions accessible beyond the location of action.

We contend therefore that even though it was not specifically designed to support collaboration, Elvin – and other pure notification services (?) like it – can nonetheless be very useful for awareness and interaction support. Elvin supports awareness because it allows us to augment the workaday world and give imperceptible computer-based events a form that can be made publicly available and accessible as an informational resource for perception. It also allows us to give distributed physical events a form that can be perceived beyond the local area. This public availability and accessibility of people and information can help makes awareness and interaction possible in distributed environments.

In the following discussions, we first outline the design of Elvin, its features and conceptual model and then go on to summarise related notification service work. In Section ??, we introduce three graphical Elvin clients: CoffeeBiff, Tickertape and Tickerchat. The various ways in which Elvin, via these clients, is used to augment the workaday world is discussed in Section ??. We go on in Section ?? to explore both the functional and technical reasons why Elvin has been so widely adopted and used within our organisation, looking particularly at how Elvin works for awareness. We conclude with a short discussion of future work.

¹ Distributed Systems Technology Centre

2. The Elvin Event Notification Service

The function of a notification service is to act as a distributor for descriptions of events. We define an event as any significant change in the state of an observed object and a notification as a computer-mediated description of that event.

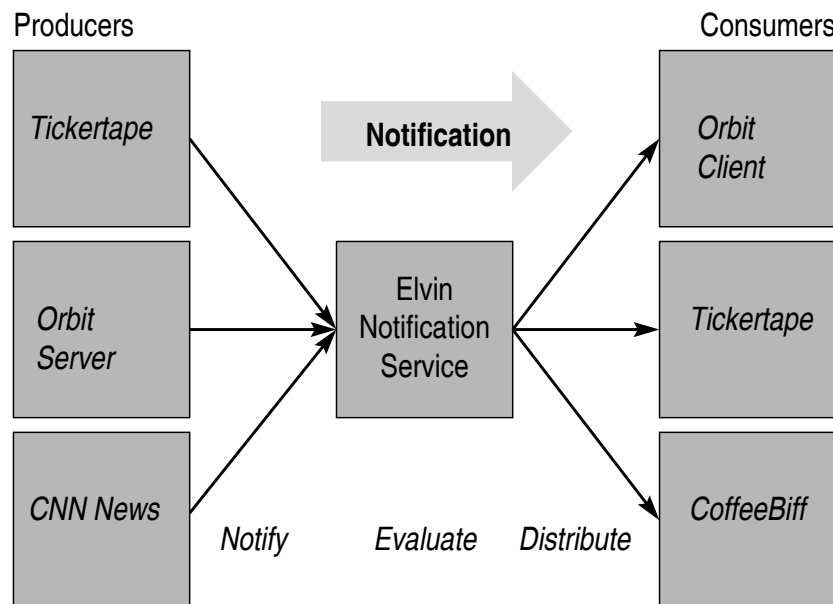


Figure 1. Conceptual Overview of Elvin.

Elvin (??; ?) is a ‘pure’ notification service (?). As illustrated in Figure ??, producers detect events (and are responsible for determining that the status change is significant), and send descriptions of the events to the service as notifications. The notifications describe events using a set of named attributes of simple data types and consumers subscribe to sets of events using boolean subscription expressions. When a notification is received at the Elvin server from a producer, it is compared to the consumers’ registered subscription expressions and forwarded to those whose expressions it satisfies.

An example of this is given in Figure ??. The producer sends the notification (shown at left) to the Elvin server. The server in Figure 2 has three subscriptions, expressed in the three boxes at right. The notification matches the middle subscription but not the top and bottom ones, therefore the server only sends the notification on to the consumer corresponding to the middle subscription (as shown by the arrow).

This *content-based selection* of notifications is often sacrificed by other notification services in favour of less flexible mechanisms because

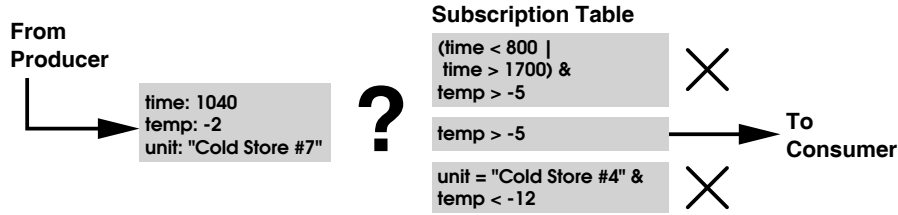


Figure 2. Content-based subscription examples for an environmental monitoring system.

it is difficult to implement efficiently. The simpler alternative is to use named channels or topics that must be specified by both the producer and consumers. A key benefit of content-based notification is the absence of this coupling between producers and consumers, promoting system evolution and integration. Elvin can still be used as a channel-based service however by selecting an attribute as a channel identifier, thus creating a channel-based service on top of a content-based service without loss of generality.

Since producers are not coupled to consumers and only need to direct notifications to the server, the determination of the significance of a state change becomes less important: they can promiscuously send notifications about any potentially interesting information, and rely on the notification service to discard those of no (current) interest to consumers, as indicated by the current subscriptions held at the service.

While large volumes of unused notifications may be useful from a user's perspective, they consume network bandwidth. To overcome this problem, Elvin includes a *quenching mechanism* that allows producers to discard unneeded notifications without sending them to the server (see Segall and Arnold ? for more detailed discussion).

In order to support organisation-wide notification, the implementation of the notification service must cater for many client applications. A single Elvin server can effectively service thousands of clients (producers or consumers) and evaluate tens of thousands of notifications per second on moderate hardware platforms. Further, additional servers can be configured in a *federation*, sharing the load of notification delivery, providing wide-area scalability (allowing Elvin to work across multiple LANs or even over the internet) and ensuring fault-tolerance in the face of individual server failures.

Like most other pure notification services, Elvin does not store the notifications it sends - it provides no persistence for notifications nor does it support message buffering (although specialised clients can be built to provide these services).

The Elvin server is implemented in C for Unix platforms, and client-side libraries are available for C, TCL, Smalltalk, Python, Lisp, elisp, PERL, and Java. The basic Elvin services are implemented via several simple API library calls that support session setup and termination, producer notification, consumer subscription and callback registration, polling (where required) and quenching.

Producer and consumer commands make the service accessible from a command line shell and from shell scripts. Non-programming users can use graphical tools like Tickertape (see Section ??) to produce and consume certain kinds of notifications.

3. Related Event-Based Work

The approaches to computer-based support for awareness are many and varied. Dourish and Bellotti (?), for example, identify three mechanisms: explicit informational awareness mechanisms (such as those provided by version control system annotation); role-restrictive mechanisms (such as those used in some group editing systems and later in many workflow systems); and shared feedback mechanisms (where information about actions or events is collected and presented as background information in a shared workspace). More recent research has, as observed by Sandor et al (?), evidenced an overwhelming concern with the latter event-based shared feedback approaches to awareness, which are more flexible than role-restrictive approaches.

Previous work on notification services for awareness within the CSCW community has tended to focus on the support of synchronous collaborative work. The term ‘notification service’ is used to refer to a variety of systems with very different behaviours and which may provide notifications directly to users or only to applications or both.

Ramduny et al (1998) introduce a taxonomy for characterising notification services based on their communication behaviour and the level at which they operate (system or user). The authors distinguish between services primarily on how they enable a client that effects a change in some piece of data (the active client) to communicate information about that change (the notification) to a client that wants to know (the passive client). They also discuss the possibilities of differences in pace and volume between system level services and corresponding user level services.

In the terms offered by Ramduny et al, Elvin is a pure notification service – active client tells notification service, notification service tells passive client – which remains completely separate from the observed data. Elvin was primarily designed to serve the system level, offering

support for applications to exchange notifications but little explicit support for user notification and volume or impedance matching; this is left as a client application problem.

Perhaps the best known CSCW notification service is Lotus Placeholder, which is based on Notification Service Transfer Protocol or NSTP (?). Rather than providing a pure notification service, the designers of NSTP opted to focus on facilities for synchronous collaborative applications. They therefore conflate a notification service with a centralised data store for shared data. The work primarily focuses on providing a protocol (based loosely on HTTP) for inter-operation between notification services. Placeholder is a sample implementation of such a service. The notification service includes a number of design notions such as, Things (roughly, application objects) in Places (generalisation of application session), with Facades (which moderate access to Things). The service is also envisioned as providing some system and some user level services ('place browsing' allows users to move between Places).

The design of NSTP means that Placeholder will primarily be useful for the construction of bespoke synchronous collaborative applications. The explicit centralisation of shared data makes it difficult to integrate Placeholder with existing applications. The complexity of the required implementation also makes it difficult to produce competing servers for Placeholder to inter-operate with. In contrast with Elvin, the service is essentially channel-based and does not appear to be scalable.

AREA (?) provides cross-application awareness support. Unlike Elvin, and in a conceptual evolution of the NSTP protocol approach, AREA requires that considerable domain knowledge be explicitly embedded in order for the tool to work usefully. Such knowledge has the advantage of facilitating support for semantic-rich notifications, but at the cost of a higher overhead of adoption and a more brittle, less evolvable system because explicit updates are needed to reflect changing application needs.

Hall et al (?) also focus on the design of a shared data communication service for synchronous groupware with CORONA. The designers of CORONA, however, partition the system into a number of services and maintain the 'publish-subscribe service' as a pure notification service using a channel-based approach. Optimised for wide-area use, the publish-subscribe service multicasts published notifications to distributor nodes which in turn multicast to other distributors which then send on to local subscribers. This enhances scalability by "minimizing system-wide awareness and change". This use of multicasting is not so easily available to systems such as Elvin because they do not rely on channel-based subscription.

Lövstrand (?) and Gaver et al (?) describe the Khronika system which, for several years, was in use at Rank Xerox EuroPARC (now Xerox Research Centre Europe, Cambridge). Khronika was fundamentally an event database that stored user-level events (meetings, brown-bag lunches, etc.). Users could discover events by browsing the database or by assigning ‘event daemons’ to issue notifications when certain kinds of events triggered.

This system is very relevant to our discussion because it had a constraint language allowing users to specify quite complex subscriptions to events in which they were interested. We can consider Khronika’s constraint language analogous to Elvin’s subscription language. Khronika is also interesting because it was primarily a user-level notification service not a system-level service. While designed as a low-level system service, Elvin can also be used for user-level applications.

NESSIE (?), like Elvin and Khronika, provides an application independent generic infrastructure. It is unclear if server efficiency and scalability have been as clear a focus of the design of NESSIE as they were for Elvin. Whereas Elvin started as a generic notification service and its application to awareness came later, as reflected in the interfaces of its clients, NESSIE was conceived explicitly as an awareness service, and substantially more attention has been given in NESSIE both to ‘physical world’ sensors which feed into the notification streams, and to various kinds of advanced, VR-style interfaces for visualising the notification information. Both of these represent directions which will also be explored in future work with Elvin and its clients.

4. Client Interfaces

While Elvin was designed as a generic notification service to be part of low-level systems infrastructure, the development of a number of GUI clients have made the functionality of Elvin accessible at a higher user-level. We introduce three interfaces here: CoffeeBiff, Tickertape and Tickerchat.

4.1. COFFEEBIFF

The simplest interface to Elvin is CoffeeBiff, shown in Figure ?? . When people go to the kitchen for a coffee break, they can click on the CoffeeBiff icon to indicate their intention to colleagues. The number on the icon is then incremented by one and their name is added to the scrolling list. A background application subscribes to CoffeeBiff notifications and sends a second-level notification to the Tickertape

‘coffee’ group (to be discussed below) when more than five people are drinking coffee, indicating that some kind of party is clearly in progress.



Figure 3. The CoffeeBiff interface.

4.2. TICKERTAPE

Tickertape (??; ??; ?) was a client interface that the developers initially created to give a visual display of event traffic and has since proved to be the most critical interface to Elvin. It is a highly tailorable window that can be used to both produce and consume notifications as desired: it displays notifications that the user subscribes to and it can be used to construct chat-style messages that are sent as notifications.



Figure 4. The TickerTape interface with scrolling message.

The typical TickerTape interface, as shown in Figure ??, consists of a single re-sizable rectangular window, showing small colour-coded messages that scroll from right to left. Each message corresponds to an Elvin notification of a specific format that has been received by the TickerTape application. For example, the left-most message in the figure is from user ‘arnold’, has been sent to the group ‘b&d’, and has the text ‘so my monitor works’. The TickerTape is designed to take up minimal space - the active area is a single line, and borders, colours, etc., can be tailored to make the TickerTape ‘fade into the background’. Most users position their TickerTape(s)² on the edges of their screens, where they provide a simple kind of peripheral access to the information that scrolls by.

Tickertape users subscribe to messages at two levels: they indicate the producers or ‘groups’ they are interested in, where group is an

² Some people choose to run multiple Tickertapes, each customised to different types of information and scrolling at different speeds.

attribute contained in the content of all messages to which Ticker-tape can subscribe, and they indicate some filters over the content of messages which have the appropriate group attribute values. A simple set of dialog and menu list boxes is used to allow non-programmers to perform this customisation. If events have a MIME attachment, associated graphics on the scrolling message indicate this, and the user can trigger the attachment with a mouse click.

Individual notifications have a lifetime over which their appearance fades from colour to grey, thus providing an indication of the timeliness and age of the information. The lifetime is user-defined for each group. Users can also choose to delete or save a scrolling message by clicking on the message itself. Tickertape therefore provides users with a mechanism for controlling the transience of information.

Earlier versions of Tickertape provided no inherent persistent storage of notifications – once a message fades away, it is lost – however notifications could be archived externally as now happens via a web-based record. More recent versions of Tickertape also support an optional auditory cue that users can select for groups of higher priority.

4.3. TICKERCHAT

Tickerchat was constructed out of frustration with the difficulty of carrying out more complex conversations with the scrolling, time-fading interface of the Tickertape. Tickerchat provides an interface for the same notifications as Tickertape in the style of the now-familiar chat tool. Several implementations have been constructed by different developers.

All of these Tickerchat implementations provide the same basic interface: a large window in which previous messages are shown, a line at the bottom in which one may enter new messages, and some way of selecting the group to which one's messages should be directed. Messages shown in Tickerchat do not fade, they simply scroll up the transcript window. Users can choose to clear or save the buffer when they no longer want to see the message stream.

The Tickerchat tool extends the lifetime of the information in Tickertape making it more convenient for discussion as opposed to notification. This semi-persistent quality allows users greater discretion about when they attend to discussions because they know they won't miss anything through time-out.

Many users choose to run both Tickertape and Tickerchat, using the Tickerchat when they enter a discussion and reserving the tape for observation of the flow of information.

A more recent version of Tickertape, shown in Figure ??, starts to mix both sets of features with its threaded history list (where the

indentation of a message beneath another message indicates that the former is a reply to the latter); this was not available at the time of our initial study.

4.4. EXTERNAL AND INTERNAL EVENT STREAMS

These three interfaces give users access to Elvin for events that can be produced both externally to Elvin and internally to Elvin³.

External system events were the primary intended use for Elvin as a low-level service. The GUI interfaces now enable access to a whole range of other pre-existing event streams. Various developers around the organisation have written programs to instrument existing sources to produce notifications that are sent as a uni-directional event stream to the TickerTape window. Examples in use here include system file changes, code debug output, and file check-in to a CVS⁴ repository. Other event sources in regular use include the following: postings to Usenet news groups, commercial news sources on the Web, personal email, a rooms booking system, and a weather station. A secondary level of notifications can also be generated, such as who has had the most cups of a coffee each day from analysis of CoffeeBiff interactions.

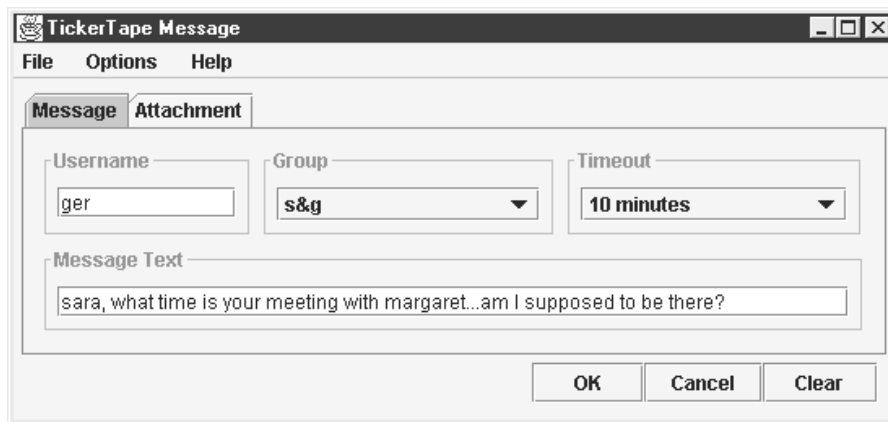


Figure 5. Generating an internal event by typing in a message in the dialogue box and sending it to a group.

Almost as a by-product of the generic nature of Elvin and the GUI clients, users can also be direct sources of events, facilitating chat-like functionality. Users define chat groups by agreement with their peers.

³ In previous discussions (?; ?; ?), we have also referred to external and internal events as uni-directional and bi-directional events respectively.

⁴ The Concurrent Version System (CVS) is used by developers to manage multiple versions of source code.

A user generates ‘an event’ internally when they either click on the CoffeeBiff icon or type a chat message into the Tickertape or Tickerchat dialogue boxes, as shown in Figure ?? . This creates a bi-directional event stream where the user can both ‘produce and consume’ events in the form of sending and receiving coffee clicks or group-based chat messages.

5. Experiences Using Elvin in the Workaday World

Via these GUI interfaces, Elvin quickly began to be used by a number of different research and prototyping groups within our organisation. The Tickertape interface in particular was the key trigger in facilitating this usage because it enabled people to quickly understand what Elvin was, how it worked, and how it could be used for their own purposes.

What is particularly interesting is that the adoption and adaptation of Elvin via these GUI interfaces happened at the grass-roots. There was no management mandate for the system to be used. There was no mandated effort to facilitate its adoption nor to promote its potential for interaction and awareness. The developers of Elvin, however, were committed to supporting users and were active and enthusiastic proponents⁵ of the value of Elvin development as a research programme.

Hence, the usage examples that we discuss here were thought of and/or developed by a wide variety of people, many of whom were not directly involved in the Elvin project. Most of the applications and tools take advantage of the Tickertape interface. Others use Elvin directly, plugging it into some other piece of software. Some are primarily to support ‘individual’ tasks. Others are to support collaborative aspects of work. Some started out to support an individual and were soon found to be useful as a way of providing others with the information they needed to better coordinate their work.

In the following sections, we briefly discuss the methodology for this study before going on to illustrate some of the many ways in which Elvin has become embedded into the workaday world of its users.

5.1. METHODOLOGY

The following comments are drawn from two studies of Tickertape use within our organisation. One was a Tickertape usage study reported in (?) (relying on interviews with 23 people, 20 of whom were current users and 3 past users, and some analysis of a log file of 20,000 messages

⁵ Some would say proselytisers!

collected over a three month period). A further survey and round of semi-structured interviews were conducted at the time of writing. The authors have also been participant-observers as regular users of the system from the beginning.

In our initial study, the average number of Tickertape messages handled by the notification server was in the order of 365 per day⁶. The average number of messages per day has risen slowly but steadily since then: 558 in December 1998, 1072 in July 1999, and 1590 in February 2000. There are currently over 40 users.

While we did undertake some basic statistical analysis of the log file contents⁷, it is the qualitative interpretation of the data that we present here for the purposes of highlighting the emergent experiences in using Elvin.

5.2. USE STORIES

In this section, we will give examples of how Elvin via its various client interfaces has been used, namely: to support informal interaction, to give digital actions a visible form, to make transient physical actions available beyond the location of action, to support content and socially based information filtering, and also as part of a general collaboration environment.

5.2.1. *Supporting Informal Interaction*

One of the most popular uses for Tickertape and Tickerchat is as a lightweight channel-based semi-synchronous chat tool. Examples of interactive chat groups are: the ‘Chat’ group for all the people in the organisation (used as a general discussion forum and for user-generated announcements), the ‘lunch’ group (used by regular lunch-goers to organise lunch times and venues), the ‘b&d’ group (for Bill and David who are working closely together on a project) and the ‘elvin’ group (for the developers of Elvin).

These groups are used extensively within the organisation by technical and non-technical staff. They support both work-related discussions and general social banter. They are also used for a variety of other purposes. *Timely announcements* of transient importance, such as “there

⁶ Sample averages were calculated from the logs for each month. The February figure, for example, represents the average number of total messages per day sent out to groups for the sample period 24-27 February 1998. The March 1998 average was 362 messages per day. Any individual would only be subscribed to some subset of the total messages sent, depending on the groups to which they subscribe.

⁷ For example, the analysis included average length of message and length distribution, number of messages per group, number of messages sent per individual or source.

```

Chat: dooley >> Can anyone help me with a query about
"H0t meta" ?? Pls ??
Chat: sjm@hq >> Try Nigel.
Chat: sjm@hq >> I think...
Chat: woody >> try Hoylen or Zhimin - not Nigel
Chat: sjm@hq >> Oh. Ok...I'll shut up now.
Chat: oracle >> woody knows all
Chat: oracle >> ask woody -- he wrote it all
Chat: sjm@hq >> Or at least more than me.
Chat: phelps >> Besides, Nigel's having coffee right now
and shouldn't be disturbed...
Chat: dooley >> Thank you all

```

Figure 6. A Tickerchat transcript of a request for help to find the ‘expert’. Nigel is at a different site to the other participants.

are cakes in the kitchen” are often made. People also use it as a type of *locator service* with messages such as “paging Dr Tim”.



The image shows a window titled 'XTickertape' with a single line of text: '? Chat:andry:Or how can I type è ê or ç in Linux(PC) using emacs ? Chat:povey:Do C-h a cor'.

Figure 7. A request for technical help via Tickertape.

Tickertape is widely used for *asking questions and getting help*, as illustrated in Figure ?? where the receptionist is trying to find the best person to take a phone query and people from three different technical groups respond. It is also used extensively for more technical requests, an example of which is shown in Figure ?. In this case, it is very similar to the experiences reported about the Zephyr Help Instance (?) and the BABBLE system (?). It provides a light-weight mechanism for accessing expertise within the organisation, but does not place directed demands on individuals; people are free to respond or not. Bradner et al ((?)) call this “unobtrusive broadcast”. Onlookers can also indirectly learn from the given answers or at least become more aware about who knows what.

Interactions over these bi-directional groups tend to be spontaneous, short, informal, often irreverent, and bursty; this too is similar to the style of interactions reported in Ackerman and Palen (?). They incorporate some of the synchronicity and immediacy of the telephone with the asynchronicity of email but where attention to the postings is entirely discretionary.

Very quickly, Tickertape has become embedded as just another means for communication and interaction along with the telephone, email,

face-to-face discussions, and porthole video images. It is not uncommon to see a discussion over Tickertape that ends with a comment such as “uh oh, see email...” or “wait a minute ... I’m coming down” when the content of the discussion becomes more detailed than can be usefully handled in short chat messages. People also use Tickertape to establish if and when a person is at their desk so that they can coordinate a phone call or a visit.

5.2.2. *Making Digital Actions Visible*

Besides chat, Tickertape is widely used to give visibility to computer-based events.

There are many examples of users writing code to generate events that reflect their activities in the digital computing environment. One user, Tim, for example, has written scripts to generate a notification whenever he logs into or out of his workstation. In this way, others know when he has arrived at or leaves from work, making both his physical and digital presence available.

There are also ‘file-watcher’ and ‘web-watcher’ event generators that monitor changes to the network file stores and local web pages respectively and send out notifications via the relevant Tickertape groups. These serve as an important information source for people who rely on system files, maintain sections of the company web, etc.

Analogous information is available via notifications generated from CVS, as will be illustrated in Figure ?? . Grinter (?) has already reported on the important role played by configuration management systems such as CVS in the coordination and articulation of software development. Elvin via Tickertape significantly enhances this role by providing a mechanism by which the development team can be notified in a much more timely way about relevant changes. The log file still exists for later reference, but awareness of changes is no longer solely dependent on team members explicitly reading the log file.

5.2.3. *Extending Information About Transient Physical Events*

In its many different forms, Tickertape is particularly useful for presenting temporally relevant information of transient importance. As a trivial example, an outdoor weather station has been instrumented to produce hourly notifications about the current temperature etc.

CoffeeBiff provides a way for another type of physical event – going for coffee – to be ‘visible’ beyond the local environment. This has had the effect of supporting face-to-face interactions; if a person knows when her friends from other parts of the building are taking a break it can help her to coordinate her break times, or to identify when a colleague will be in the kitchen and possibly free for a casual chat.

A more useful example of this feature is the instrumentation of the DSTC Rooms Booking calendar. At ten minutes prior to the booked time, a Tickertape notification is generated stating the time, room, and meeting description. This is similar to the uses of the Khronika system (?).

5.2.4. *Supporting Social and Content Based Filtering*

Elvin via Tickertape is frequently used to support information filtering based on *content*. This means, for example, that a user can essentially say “show me all the postings to comp.groupware that mention ‘shared drawing tool’ ”.

People have also used the content-based subscription feature to implement a form of *social filtering* of information. For example, several people have subscriptions which essentially say “if this person from our organisation posts to the (newsgroup), tell me about it”. This is making use of what they know about others’ shared interests and areas of expertise — it is highly likely that if that person is contributing to a thread, then it will be of interest to them as well.

There are also several ways for users to be alerted via Tickertape that they have received *e-mail*. Users can add a script to their mail processing (.forward) file which generates a notification with appropriate summary information (sender, subject, and MIME information), and sends it to a private Tickertape group. Users can also filter mail notifications on content and so receive notice of only the important or interesting email.

5.2.5. *Using Elvin in Orbit Collaboration System*

Although we described Elvin as a generic notification service rather than a collaborative application service, it can still be used in collaborative environments as we have done with Orbit (?).

Orbit is a collaborative desktop environment that has a client-server architecture. When a client initiates interaction with the server, it uses a remote procedure call (RPC) mechanism (CORBA). When the server communicates with a client, it is because the client needs to know that server’s state has changed. By sending a notification via Elvin about state changes, the Orbit server is able to offload the responsibility of keeping track of which client is interested in what information.

Tickertape has also been extended so that interactive groups can be created that correspond to an Orbit group zone, thus providing Orbit users with a low-bandwidth communications tool that is specific to their work context. In this way, Elvin is being used to support both low-level and user-level needs simultaneously.

5.3. USING ELVIN: SUMMARY

In this section, we have shown a variety of ways in which Elvin has been used within the organisation. Many of the cases given above are not intentionally CSCW-related, and this is precisely what makes them interesting to us. They are all examples of ever more powerful facilities being evolved through the incremental instrumentation or augmentation of workaday tools to make information and people more visible, available and accessible.

The chat-based tools and CoffeeBiff tool have been significant additions to existing communication resources for supporting interactions, both for work and fun, in a distributed workplace. Not only have information and expertise been shared, but social cohesion and relationships have also been strengthened both within sites and across sites.

The uses where Elvin, mostly via Tickertape, has been used to push information to users from externally produced event streams has radically changed the ways in which people go about accessing and using many of these external sources (such as email, WWW, Usenet news etc.). Other uni-directional uses, for example, where CVS or the Room Booking systems have been instrumented, have facilitated opportunities for far greater visibility of activities and events both within the digital realm and the social/organisational realm than has otherwise been possible. Opportunistic discussions can also take place around the work event.

It is in the integration of all these facilities that the value of Elvin and its clients becomes obvious for enabling greater collaboration in the workday world. The following is one example.

5.3.1. *A Late Night Discussion around CVS*

The dialogue shown in Figure ??, is an actual experience by the developers of Elvin around their use of Elvin-instrumented CVS.

It is late at night and Phelps is working from home fixing a bug in some Elvin code. When he is finished, he checks the file back into the code repository using the version control system, CVS. When prompted, Phelps enters the comment “The gap doesn’t actually need to draw anything...” into the attached log file and completes the check-in. The developers have instrumented CVS so that this event causes an Elvin notification to be generated stating the name of the modified file, in this case “gap.c”, and the associated comment. The notification is then sent by the server to people subscribed to the ‘elvin’ group. David (‘d’) is working late back in the office and sees that Phelps has made some changes to the ‘gap.c’ file (history line 1). He sends a message joking “and you tell me to

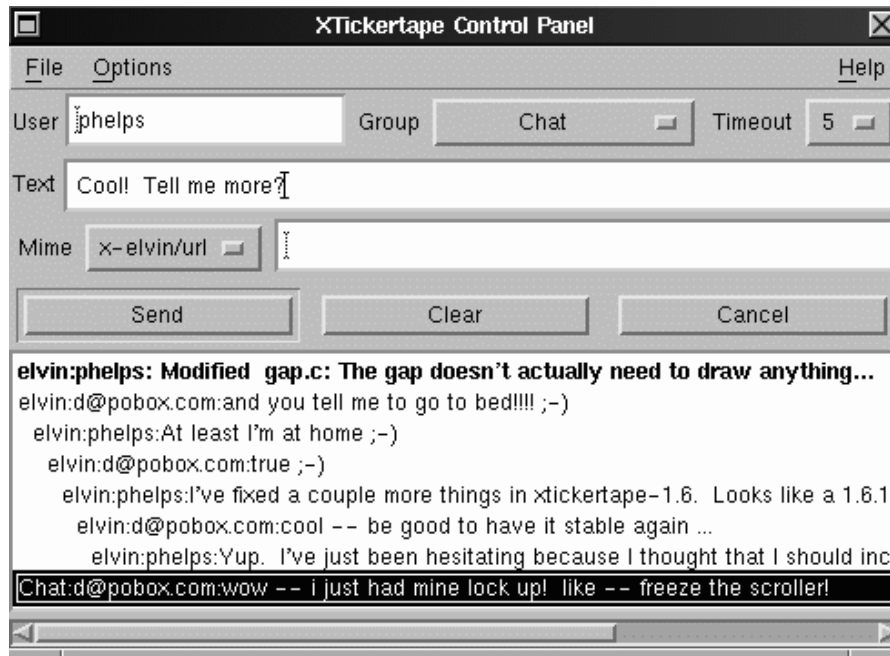


Figure 8. New version of Tickertape with threaded history dialogue showing a discussion generated by a file check-in to CVS.

go to bed!!!!;-)” (line 2). Phelps and David then engage in some light-hearted banter about their working habits (lines 3-4). Phelps goes on to explain a bit more about what else he has been working on with a version of tickertape and they have a short discussion around that work (lines 5-7). In the middle of the discussion, David has a problem with his tickertape that illustrates the bug Phelps has been trying to fix: “– i just had mine lock up! like – freeze the scroller!” to which we see Phelps starting to respond in the dialogue text box “Cool! Tell me more?”. And so they continued to discuss the new problem.

This is a compelling example that reflects a type of experience that can happen many times per week, especially for the members of this group, and even more so recently since two members of the team have re-located to another city. Via the one interface, Phelps and David have been able to find out about computer-based events as they happen, engage in social chit-chat, have a timely work discussion, and ‘be there’ when a problem happens to engage in collaborative diagnosis. They did not have to go to separate tools for notification and for chat. They did not have to forgo their preferred work environments. Elvin via

Tickertape integrated with, and augmented, their workaday worlds to add another layer of information and communication support.

6. Reflections on Why Elvin Works?

In this section of the paper we turn our attention to the question of why Elvin and its graphical clients have proven to be so successful and how it is that they provide support for interaction and awareness. The Elvin/client suite is certainly not a well crafted CSCW application, based on careful and principled user study, designed with user participation, and so forth. Elvin was designed primarily as a good generic event notification service; yet it has proven to be remarkably useful, adaptable and robust, both technically and in terms of the kinds of uses to which it has been applied.

The essence of the success of Elvin and its clients is that they fulfilled a key need: providing a simple way to gather and redistribute information produced during the everyday work of our users (via Elvin) and to give that information timely perceptible form (via the various clients). Importantly, they did not detract from nor inhibit the use of existing applications and tool sets. At the same time people were able to flexibly, spontaneously and incrementally construct a suitable workaday environment that afforded substantially improved support for awareness and interaction via a single unobtrusive interface that could be turned to any number of purposes. It is for this reason that we talk of Elvin ‘augmenting the workaday world’ (?)

Another important factor was the willingness of some of the more technically-inclined users to develop different client interfaces or to write the scripts to generate the notifications in the first place. It is significant that no one was asked to do this work, rather people voluntarily and spontaneously went ahead with it because they believed, and indeed found, that the facilities would be useful for their everyday work (thus avoiding Grudin’s (?) work-benefit disparity problem) and then willingly shared the results with colleagues, thus participating in the evolution of system use.

We contend that there are both functional as well as technical reasons for this success. Both are critical; a complicated technical tool providing the same functionality is unlikely to have been adopted by the user community, similarly with a simple tool offering limited functionality.

In the following discussions, we describe the *functional* reasons for Elvin’s success in terms of how it provided available and accessible information as a resource for awareness via the graphical clients and

how it is that these are integrated with interaction. We also discuss the complementary *technical* reasons, looking at the consequences of the conceptual simplicity and generality of the design of the Elvin server, and at the critical role played by the Tickertape interface in its adoption and user-accessibility.

6.1. PROVIDING INFORMATIONAL RESOURCES FOR AWARENESS

Elvin provides this support for awareness and interaction by gathering and presenting information in a form that enables public *availability* and *accessibility* and where multiple forms of information and interaction can be *integrated* in a way that re-connects bodies with actions.

6.1.1. *Making Information Available*

When we are co-located, a large part of how we maintain a sense of who is around and what is going on is by being able to see and hear events or actions such as people arriving or leaving, phones ringing, sighs of frustration, conversations, the grind of the coffee machine, and the hum of the printer. These are all things we can potentially use to coordinate our work and play together and we often do so with little conscious effort, as well demonstrated, for example, through the work of Heath and Luff (?).

In these co-located situations, where we are bodily immersed in the space and where the laws of physics prevail, it is easy to overlook the fact that we can hear or see (or touch or smell or taste for that matter) because there is something to *be* seen or heard. The actions or events that we just talked about create effects in the physical world that can be mapped down, for example, to sound waves or light patterns, and so on. These effects become the publicly-available informational resource for perception, and hence create a possibility for awareness, which in turn can enable communication/coordination among people. As stated by Robertson, “For participants in a cooperative process to be aware of anything, it must be publicly available to them. And the public availability of anything is dependent on its perceivability.” (?).

When people are distributed we lose access to this information because the informational effects are only available for local perception.

In the digital computer-based world, there are also activities that could potentially be useful for maintaining a sense of who is around and what is going on: people log in, files are opened, changes are made, email is read and so on. Further there are a whole range of other ‘events’ that are computer-initiated (though effected by a human-program), such as a change in the value of a variable, that could be useful to know about.

These activities and events do not ‘naturally’ produce effects that we can perceive at the moment of action in our physical world: “people and physical space may be made of the same stuff, but people and virtual space are not” (?). That is, there are currently no inherent mechanisms in the digital realm by which these actions are publicly available in a form that people can perceive. Even the individual’s perception of their own action is mediated indirectly via generic key press actions on a keyboard, with possibly some visual feedback depending on how the environment has been programmed for that task.

An Elvin notification provides a way of capturing the fact that a computer-based event or action, such as Phelps checking-in his file, has taken place. Through a client interface such as Tickertape, it can be given *perceptible form* as a Tickertape text message. This representational form is then publicly available to all who have indicated an interest by subscribing to that group. Again from Robertson (*op.cit.*): “[it is this] public availability [that] defines and enables the communicative potential of actions and artefacts within any shared environment”.

The communicative potential of publicly available actions is evidenced in discussions with some of the developers who use Elvin-enabled CVS. They report that their log comments are now very much, as described by one person, “active communicative messages more focussed on the short term” whereas previously they had only thought of them as providing an historical record for their own use. This consciousness of the communicative nature of their actions is also evidenced by the increased frequency with which the developers are checking-in changes: “we want to be seen to be doing work” – an example of how individuals “produce particular activities to make them selectively visible to others” (?) in this digital domain.

The CVS example of Section ?? also illustrates that Elvin only provides a *resource* for awareness and that having an ‘other’ *perceiver*, is an important part of realising the potential of this communicative action. While many people were subscribed to the ‘elvin’ group, Dave was the person who happened to see the message from Phelps’ action and respond to it. As stated by Heath et al (?), “awareness is accomplished both by the ‘recipient’ of particular activities and the individual who produces them”.

One limitation of *physical world artefacts and actions* is that the public availability of information about them is only accessible to those who are co-located (within some appropriate definition of co-location) – hence the strong interest in the CSCW community of how to support people who are physically distributed and cannot directly perceive actions, artefacts and people in other spaces. Our DSTC community is distributed in a variety of ways across offices, floors, buildings and

cities. As such, people cannot ‘see’ when a colleague in another city, or indeed even in another office, arrives at work or goes for coffee and so on. Elvin via its clients provides ways to give representational form to such physical world activities so that their potential as a resource for perception and awareness can be made available beyond the local place of action.

For example, the Tickertape notifications generated from the web-based Rooms Booking System gives everyone a chance to know what sorts of meetings are going on in the organisation, who might be visiting, etc., even if they are unable to wander past a room and glance in. Shared CoffeeBiffs across sites are even useful: in the exchange illustrated in Figure ??, Nigel is in a different city to the other participants yet Phelps could still comment that “... Nigel’s having coffee right now ...” because Nigel had previously indicated this via a cross-site CoffeeBiff and Phelps had seen it.

These experiences are similar to those found with the Khronika system: “It enhances our general awareness of ongoing events and this promotes collaboration. It does so in a way that blurs the boundaries between the electronic and everyday worlds, allowing information to be entered from and disseminated by both.” (?).

6.1.2. *Making Information and People Accessible*

Elvin via the various interfaces also enables easy and timely accessibility to both information and people.

Information sources such as usenet news, the WWW, email, CVS logs, etc., are already accessible via the ‘usual’ mechanisms, e.g., running a news-reader or web-browser or mail application, opening up the log file, etc. The content-based filtering capability for producing Elvin notifications provides another avenue of access to this information by enabling a different representational form and bringing the information into the ambient background of the user’s work environment via the client interfaces. In this way, the information is more immediately accessible because it is brought ‘to’ the user. It is also likely to be more timely as the information is available at the moment of change or posting, not when the user gets around to reading the source. These findings concur with those of Fussell at el (?) who suggest that “awareness tools, then, may benefit users when they provide passive awareness of information that previously had to be actively sought.”.

People are already accessible in a variety of ways, via phone, email, face-to-face interaction, etc. The open light-weight chat-like facilities of Tickertape and Tickerchat provide another avenue of access. These facilities help to decrease the effects of physical distribution and start to enable some of the affordances of shared offices where people can

do the equivalent of ‘turning around’ to ask a question or bump into people in the corridor. The ease with which someone can ‘be present’ in a lightweight form and the ease of accessibility promotes opportunities for informal serendipitous interaction.

6.1.2.1. *Related Chat-Like Experiences* Our experiences about the usefulness of a chat-like tool for the support of interaction at work are very similar to those reported elsewhere.

In its role as help-tool, the experiences with Tickertape/Tickerchat are similar to the Zephyr Help Instance (?); except that Zephyr is a dedicated help tool and also supports an archived FAQ.

In its role as a general chat tool for a range of uses, the experiences are also very similar to those reported about the use of BABBLE (?); except that BABBLE is a dedicated chat tool with enhanced features such as the graphical social proxies showing the ‘location’ of users, and the user lists. It also supports separate threading of chat by topics whereas Tickertape and Tickerchat interleave topic groups.

As a textual chat facility, Tickertape and Tickerchat are also similar to a MUD but without the overhead of the descriptive MUD world. While our tools provide affordances similar to those of a MUD, as reported by Churchill and Bly (?), we have avoided some of the pitfalls that they describe, notably the problems with participating in multiple ongoing chats simultaneously, and the disconnection between the MUD and the user’s real-world tools. Tickertape/Tickerchat enable simultaneous access to multiple channels via the one interface. The disconnection problem is lessened by being able to integrate Elvin with everyday tools such as CVS, and having the Tickertape/Tickerchat window always opened at the edge of the screen.

Tickertape/Tickerchat are also similar to many of the widely-used Instant Messaging (IM) tools such as ICQ or MSN Messenger. Because both tools are graphically small and unobtrusive, people seem content to leave them running on their screens, so both afford casual contact between individuals. The main difference is that, because IM tools focus mostly on one-to-one contact, they do not support the kinds of casual multi-person conversation shown in Figure ??.

Tickertape and Tickerchat lack some of the purpose-specific functionality offered by dedicated chat-like tools. Nonetheless, they still serve well enough as an interactive medium but have the advantage of being able to integrate the chat more closely with the workaday world because of their generic adaptability, as illustrated in the CVS example.

6.1.3. *Integrating Information and Interaction – re-connecting the body?*

The real value from Elvin and the Tickertape/Tickerchat clients is gained in this integration of information and interactive chat in the one interface. The following are some tentative reflections on why this is so.

Being embodied in the physical world, there is an easy sub-conscious flow from the embodied person producing information as a resource for perception, to the other's embodied perception of that information, and the acting on or responding to that information. There are numerous examples of this in the reports of Heath and Luff (?) and Robertson (?).

Event notification services such as Elvin tend to produce a disembodied discontinuous notion of action by enabling a selective capture and representation of action independent of the body (remembering the body is not made of the “same stuff” as the notifications).

Our experiences with using Elvin via the Tickertape/Tickerchat clients suggest that the integration of event notification with the ability for lightweight chat and easy access to people starts (albeit just starts) to enable actions to be re-connected again to the embodied person producing that action, at least in the mind of the perceiver – Phelps checks in a file, it is given visible form as a notification, David sees it, and then David starts chatting with Phelps. While the notification of the check-in action is still disembodied, David's response suggests that he clearly interprets it as Phelps' action, not just as the log database being updated.

We suggest that the integration of various information sources and chat facilities in the one interface starts to approximate the easy flow that happens in the workaday world, part of the inherent appeal of Elvin and its client suite.

6.1.4. *Interesting Differences*

There are some other subtle but important differences here between public availability and accessibility in the physical realm and as mediated by Elvin; we've already mentioned the embodied nature of action.

Order of effort for awareness is one. Phelps and Dave had to undertake explicit actions of subscribing to the group, of running Tickertape in order to have the potential to perceive that event, and then of attending to the message when it scrolled past. No such explicit effort is required in shared physical space environments for potential perception – we can't help, for example, but be 'see-able' if we are co-present with another. Instead, we tend to expend explicit effort to limit or filter

perception, e.g., by putting in ear plugs so as not to be distracted by noise or shutting the office door to avoid being seen.

On the other hand, the order of effort over the long term for information filtering is lessened. Having set up an initial subscription, many people report that they rarely go directly to news or the web now unless a message on Tickertape has indicated an article of interest. People also reported being far more selective about when they read email as the Tickertape notification helped them prioritise emails.

Choice is another difference. One of the concerns raised by Hudson and Smith (?) regarding systems supporting awareness is the tradeoff between privacy and awareness. With the way people use Elvin here, there is a degree of choice about making information available that they do not have with the physical world. For example, Tim chose to write the scripts that generated a notification whenever he logged into or out of his workstation. People can choose not to click on CoffeeBiff when they are busy and want to get a cup of coffee as quickly as possible.

Hudson and Smith (?) also raise concerns about the tradeoff between awareness and disturbance and suggest that “for systems of this type to work well, it is important to place at least partial control of overt interruptions in the hands of the receiver of the information” (citing (?)). Fussell et al (?) are similarly concerned with the attentional overload of maintaining passive awareness. With Elvin and Tickertape, users have control over what groups they subscribe to, how long the information is available for, and whether or not they even have the Tickertape window open.

The culture of use around Tickertape is strongly *discretionary* and *non-essential*, meaning that “it’s OK to miss things”. People can choose to read notifications or not. They can choose to indicate they have ‘heard’ a conversational turn by responding or not. The ambient nature of the presentation matches well with this non-essential use culture. Tickertape therefore turns out to be most useful for relatively ephemeral, temporally-relevant information or when there are other ways of accessing the information if a message is missed (such as visiting a web site or looking at the CVS log) to recover what’s missing.

The number of *channels* available for providing perceptible information is another difference. The current Elvin clients only enable perception via limited perceptual channels, i.e., visual and possibly auditory channels. Much richer perception via all the sensory channels is possible within the physical world.

Most interesting though is the way in which Elvin notifications can change the *temporal and spatial relationships* between the occurrence of events, and when these events can be attended to and by whom. Because the notifications received on Elvin clients have user-defined

time-out periods, the information about an event can persist beyond the duration of the event itself, while at the same time still being transient. So, for example, Phelps's check-in of the file might have happened at 11pm but Dave might not have looked at the notification until some minutes afterwards. If this was a physical event where Phelps put a file into a drawer, the perceptible information about that event would only have been available for the duration of the event⁸. Normally only the people who share an office with Tim would know when he arrives at work but because his log-in generates a Tickertape message, people who are in another city can now also potentially know about it.

6.2. BEING A GOOD GENERIC NOTIFICATION SERVICE

Complementary to the functional reasons, there are a host of technical reasons that also are critical to the success of Elvin. Essentially they are all to do with the fact that the Elvin server was purposefully designed to be a 'good' *simple generic pure notification service*. The qualities of simplicity and generality are also evident in the client interfaces. Tickertape was the most critical of these interfaces.

6.2.1. *Elvin Server Design Consequences*

We believe there are several consequences from the design of Elvin that have contributed to its widespread use. These characteristics make the use of Elvin popular with authors of new software and straightforward to add to existing software.

First, Elvin is highly *flexible* and *adaptable*. As a 'pure' notification service (?), its only purpose is the notification of events. Elvin does not store nor model any application semantics nor does it have inbuilt policies about information distribution. Further, Elvin notifications are not explicitly typed. Every notification is simply a tuple of attribute-value pairs. Content-based addressing also proves to be a powerful mechanism in exploiting the genericity of the notifications.

These architectural choices make the introduction of new kinds of events a simple matter. Clients can gather any information in a notification, not just based on the sets of events the original authors thought might be interesting or useful. In this way the notification service problems, identified by Sandor et al (?), of having to pre-specify all message types are avoided. Hence Elvin can be used with potentially any kinds of events and is just as easily applied to low-level infrastructure uses as to the user-level applications reported here. Importantly for augmenting

⁸ As opposed to perception of the new location of the file which can happen at some later time after the event that put it there.

the workaday world, it can even enable the sharing of information from applications that are not specifically designed for collaboration.

The architectural choices also mean that Elvin is highly *evolvable*. It is a relatively trivial matter to add new event sources or new consumers or to expand or re-purpose notifications. Being dynamic, these changes can be made ‘on the fly’ without having to do any compilation, server re-start, or client updates. This supports piecemeal growth where the use of Elvin can be evolved as the user community’s needs evolve.

Further, the process of ‘*elvinizing*’ an application is simple because of the simplicity of its API. There are APIs available for several programming languages, all at a high level of abstraction and usually involving around five method or procedure calls. This is appealing for busy developers as it is a relatively straightforward matter to instrument applications to send or receive Elvin notifications. Indeed, the limiting factor has proven to be the target application and its lack of openness or APIs rather than Elvin itself; if an application has no source code and no APIs, it usually isn’t possible to instrument it.

Elvin is also technically ‘*affordable*’ by a number of measures⁹. It has no persistent memory nor buffering requirements as current subscriptions are processed as a notification appears. The quenching mechanism means that unnecessary network traffic is minimised making it network-efficient; users can ‘promiscuously’ cause notifications to be generated without regard to who might receive them now or in the future because they know they will only be sent when there is a subscribed interest. It is also affordable from a screen real-estate point of view because the clients take up minimal space.

6.2.2. *Tickertape Client*

Without the Tickertape interface, however, it is doubtful whether these design features alone would have made Elvin so popular among the user community. Hence it is doubtful whether its potential for support of awareness and interaction would have been realised.

A significant effort in making any notification service more accessible at the user-level is the effort involved in building an appropriate client. Despite some obvious shortcomings of Tickertape both as an application and an interface, its main feature is that it already exists and is *generically functional*. In being functional, it helps make the workings of the underlying Elvin more accessible, understandable and usable by the broader user community. Once installed then, the one client

⁹ While not a technical reason, it is also affordable from an attentional point of view, as noted earlier, because of the discretionary way in which it can be used and because of the potential availability of the information from other sources.

application can be used for a whole variety of purposes, overcoming the need to build purpose-specific clients.

Tickertape is also *highly tailorable*, in terms of presentation as well as functionality. This tailorability is accessible via a range of mechanisms supporting different skill levels. This accounts for much of the emergent use and uptake we see around Tickertape; adopting or adapting Tickertape is an incremental, evolving process. For presentation, the look and feel of the window can be changed so that it blends in with the screen background. For functionality, users can uniquely tailor the notifications they receive to suit their needs and work styles; each user of Tickertape has a different mixture of chat, information source (email, news, etc.), and access to changes in the state of their digital world (e.g., CVS check-ins) that is useful for them. Varying levels of expertise for defining subscriptions or creating notifications are supported through multiple access mechanisms. Non-technical users can take advantage of the GUI subscription editors and generic clients such as ‘xconsumer’ and ‘xproducer’. The more technical people can use command line clients or the APIs discussed previously.

Tickertape therefore provides a ‘*good enough*’ interface for *multiple uses*, from notification to chat, in a way that is easy to integrate into the workaday world of users even if it promotes less than optimal strategies. In the room booking notifications example, it would be useful to have an interface that would also allow one to cancel a booking. Our on-line notifications from the weather station could probably be better represented graphically. But because Tickertape already existed as a generic client, it was significantly easier to add the rooms booking or weather sources than build new clients – the Tickertape solutions worked well enough even if not optimally.

7. Conclusions and Future Work

The thesis of this paper has been that a user-accessible generic notification service such as Elvin can be used to both enable users to maintain awareness of the activities or status of others and support interaction between users.

Elvin provides a publish/subscribe notification service through which notifications from a producer are sent to consumers who have expressed interest in these notifications through a subscription. Matching in Elvin is content-based, i.e., the entire notification is used to determine matches rather than simply using pre-defined fields. This adds substantially to the flexibility and power of the Elvin system. Despite this additional overhead, the implementation is very efficient.

Through the evolution of various interface clients, Elvin has become widely adopted and adapted within DSTC to support a number of functions, from interactive chat, to notification of both computer-based and physical world events, to information filtering. None of these functions are novel in themselves. Nor are they all optimally supported. What is novel are the synergies that arise from the integration of these functions, (and potentially many more), through one simple interface. Thus, for example, computer-based events such as the check-in of a file can be ‘seen’ as a timely communicative action from which both a social exchange and a work-related exchange can easily flow. What is also novel is the technical simplicity and wide applicability of Elvin that makes it easy to instrument everyday tools such as CVS to generate the notification that started this exchange.

In these ways, Elvin has been used to augment the workaday world to make available ‘event-based’ cues for use as perceptual resources for awareness, and which have previously been missing in distributed or computer-based activities. As an added benefit of its general applicability, it has also facilitated lightweight informal interactions integrated closely with everyday tools and resources.

Elvin and its client applications have grown up in a co-evolutionary way, as both the technology and its users adapted to one another. It has demonstrated itself to be highly successful in our environment, providing a simple yet flexible way of keeping people aware of, and thus coordinated with, the activities and availability of others.

Our experience raises interesting questions concerning the ways in which information is used perceptually to support continual, contingent rearrangement of activities. It also raises questions about how effectively awareness information traditionally available through co-location, ‘being there’, can be translated into other media, and how the boundaries between traditional ‘being there’ and awareness information of the kinds transmitted through Elvin can be blurred together. A major point here is the distinction between ‘unthought’, ‘amorphous’ awareness through ‘being there’ and more structured, explicitly event-based and discrete awareness transported through Elvin and its clients. Can the body be better re-connected with digital or distant actions?

Another question is to do with ways in which we might move technology-based awareness beyond the explicit event-based notification of information to users – at present one event in Elvin relates directly to one notification – to approaches in which the underlying Elvin events are treated more homogeneously with the real world, for example by combining and synthesising events, or treating events as triggers for light, sound or other more ambient notifications.

In support of this, Elvin and its client suite are continually evolving. Future technical work on Elvin will focus on infrastructure issues concerned with scaling and quench propagation, secure notifications and event correlation through the generation of higher-order events from patterns of simpler events. We also plan to continue to augment the working environment through further instrumentation of applications and utilities. Much of this work we expect will continue in the tradition of projects such as Khronika (?), which also investigated the kinds of boundary-blurring discussed in the paragraph above. We also plan to begin instrumenting the physical world, through the development of sensors and actuators that can transmit and respond to Elvin events. We hope this approach can begin to more effectively integrate the digital and physical worlds.

8. Acknowledgements

Bill Segall and David Arnold are the key designers and developers of Elvin, with contributions by Julian Boot. Ted Phelps, Blaize Rhodes, Julian Boot, Michael Lawley and Shelina Gorain have all developed different versions of the Elvin client interfaces. Sara Parsowith contributed to earlier evaluations of Tickertape usage. Donagh Austin undertook the statistical analysis of the Tickertape logs producing the usage figures quoted here. Ted Phelps contributed to the development of this paper. We thank the reviewers for their thoughtful comments for improving the paper. We also thank the user community at the DSTC who contributed to the evolution of Elvin applications and appropriated it into their daily lives. More information about Elvin can be found <http://www.dstc.edu.au/Elvin>.

The work reported in this paper has been funded in part by the Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Industry, Science & Resources). This work has also been supported in part by the United States Defence Advanced Research Projects Agency under grants F30602-96-2-0264 and F30603-94-C-0161 (both administered by the US Air Force through Rome Laboratories). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, expressed or implied, of the Defence Advanced Projects Research Agency or the U.S. Government.

