Reflections on Long-Term Experiments with Public Displays

Adrian Friday and Nigel Davies, *Lancaster University, UK*
Christos Efstratiou, *University of Cambridge, UK*

A reflection on the authors' experiences with public display research—systems built and lessons learned—explores content creation and control, programmable infrastructures, and applications, to offer unique insights for those considering research or practical deployments using this technology.

Our first venture 16 years ago into public display research was motivated by a need to incentivize staff into wearing Active Badges that could track their location.

The public display system we built, called FLUMP (FLexible Ubiquitous Monitor Project),[1] consisted of two wall-mounted CRT monitors powered by Apple LC computers and provided information primarily for staff. Somewhat ahead of its time, FLUMP could show a personal contextually triggered page when it detected a user wearing a badge nearby—thus providing an incentive for users to wear their badge. Through a simple back end of marked-up pages and Unix shell scripts, users could customize their page to include the current state of their mailbox and content scraped from the early Web such as weather reports, sports scores, or a daily cartoon.

Limited interaction was possible: users could page through the content with the two control buttons on their Active Badge. The interaction possibilities were necessarily limited—for context, recall that in 1996 a state-of-the-art PC had a 150-MHz Intel Pentium processor and most people did not own a mobile phone, let alone use texting; SMS had been demonstrated only two years previously; and the first Bluetooth specification would not arrive for another three years.

Regrettably, we were forced to remove one of our FLUMP stations from a public area after a short time because of fire safety concerns—our first hint that deployment "in the wild" brought its own unique set of challenges.

From those challenges, we learned many lessons. It is our goal here to present both the highlights from our long-term experiments with public displays and to clarify what those lessons were for the benefit of other researchers.

E-CAMPUS: RESEARCH TESTBED

After FLUMP, we continued to experiment with displays. At a major middleware conference in 1998, we created a custom "tower of monitors" installation that showed a simple carousel of information about the conference. This was a single-purpose system that offered no interaction possibilities or content customization to viewers. Other researchers in our department at Lancaster University also began to conduct public display research, most notably on the Hermes system that explored the notion of semipublic displays through the creation of a system of interactive door plates.[2]

In 2004, we began our most ambitious project to date—e-Campus. This required a large (£0.5 m) investment and was intended to encourage the development of a public display research testbed on our university campus. Importantly, e-Campus was required to serve as a laboratory for other researchers and to accommodate a range of content types, from the needs of traditional digital signage images

 Published by the IEEE Computer Society

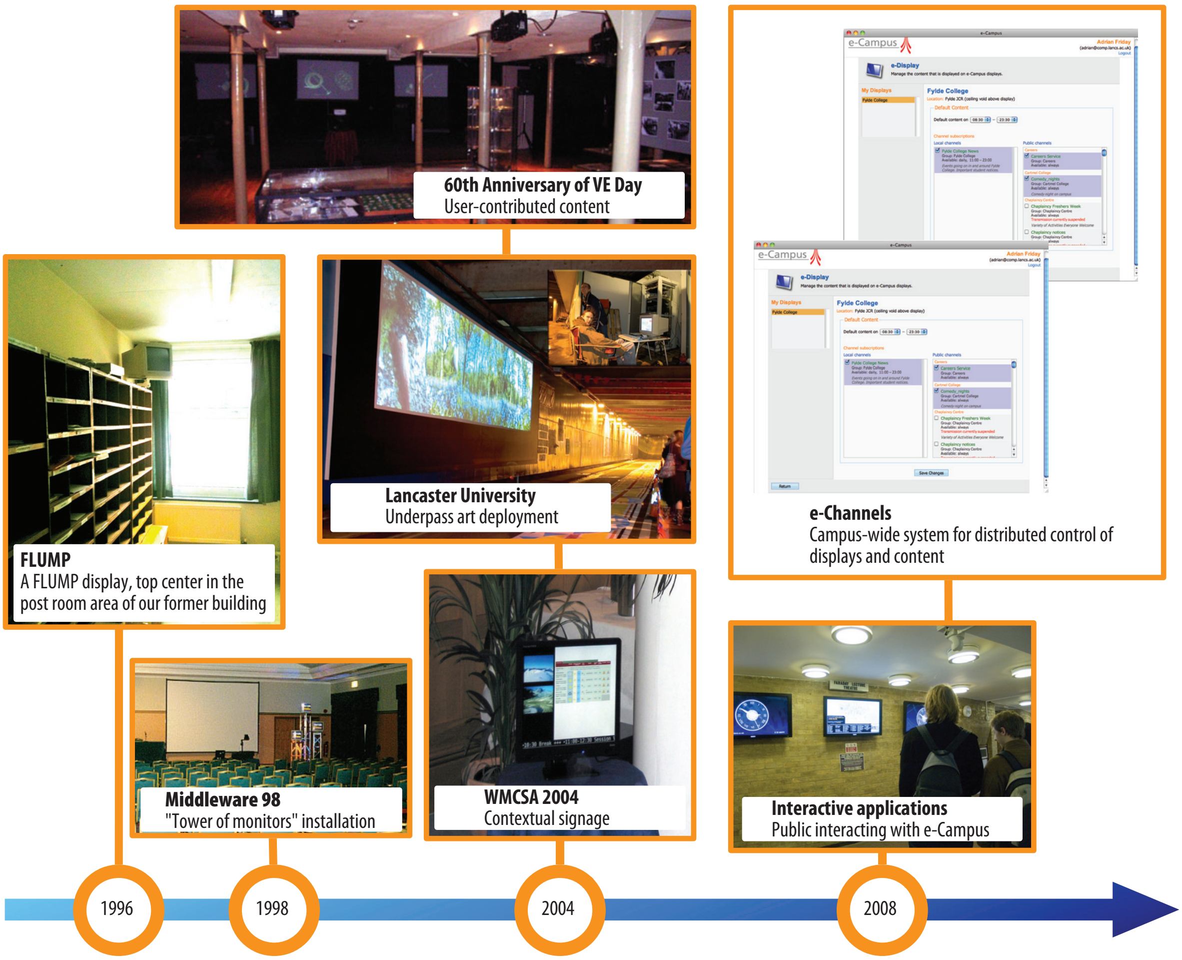


Figure 1. Timeline for experiments with public displays. (Underpass photo used with permission, copyright 2005, David Molyneaux.)

and movies to research-led explorations of interactive applications and situated games.

In evolving the design for the e-Campus software infrastructure, we began with two short-lived technology probes—one at an IEEE workshop in 2004 and another at a two-week celebration of the 60th anniversary of VE Day in a regional art gallery—and one long-lived probe, an art deployment in an underpass, shown in Figure 1. These probes helped determine the system's requirements and provided a wealth of early deployment experiences.[3]

Our original aim for the e-Campus deployment was to facilitate research in public displays. However, during the early steps of the infrastructure's installation, it became clear that even the display's physical presence triggered a desire for a system that could offer an everyday service to the campus community. This proved to be positive—it meant we had a strong user base that cared about the screens, and the fact that there was content regularly on the screens meant that we could minimize the novelty factor in our experiments.

The provision of a general, digital signage solution for the university thus became a key objective for the success of e-Campus. Since at deployment time we had no software of our own to provide signage functionality, we initially rolled out a system capable of switching, at runtime, between two distinct modes of operation: a production mode that used commercial software to support general signage requirements, and an experimental mode that used the various prototypes we built to support new types of content and applications. As our experimental system became more robust and we found that users preferred this to the commercial offerings, we gradually phased out the dual-mode system and replaced it with a system entirely based on our home-grown software.

As of this writing, e-Campus has been fully operational for seven years, has grown to 30 installations around the Lancaster University campus, and is in daily use as the main digital signage and emergency alert solution on our campus. The system has 81 individual users in 33 groups who have created 3,700 content items (amounting to nearly

5 Gbytes). In addition to its normal signage and alert roles, e-Campus has been used for arts festivals, numerous student projects, and as the basis for many research projects on interaction with public displays.

USER CONTENT AND SCHEDULING

Our research has focused on how to support users in creating and displaying their own content rather than on our creating the content. In the early FLUMP system, we required users to create their own page if they wanted it to be displayed on the screens. In e-Campus, we believed that, given the system's scale and the likely complexity of requests, users should be afforded access to tools for scheduling content on the system.

User needs

For regular contributors, we gave access to a commercial signage system with facilities for content creation, scheduling, and play-out, and for monitoring the play-out devices' software and hardware status. Content had to be arranged into multitrack timelines called playlists, and users specified which play-out devices to schedule the playlists to.

The key design insight was to separate the roles of content providers and display owners.

Several key user groups—including people engaged in marketing, some colleges, and public arts organizations—had access to this software. For more casual content providers, we built a Web-based "drop box." This simple system offered a workflow for handling requests to schedule content items uploaded to the website. The system notified us when content was ready to be scheduled and let us easily accept and reject the content. In most cases, we used the commercial signage solution to add the content to the displays' play-lists. The systems were deployed concurrently.

After 16 months, we found that the overwhelming majority of content creators were using the drop box. While the commercial signage system offered a high degree of control, using the drop box was far simpler. However, we needed to serve as a centralized administration team to handle the content scheduling, which put us in the uncomfortable position of policing content acceptability. We needed a better solution for returning control to the users.

Through a series of interviews, we found that stakeholders had diverse requirements: users wanted to control their own displays and make changes in real time; in addition, we needed a way to have content appear on other displays around campus, particularly news and events information. As we had already seen when using the commercial system, we had to balance the level of control against offering too much complexity to users.

The e-Channel system

We designed our third interface for controlling content on the public displays through a process of participatory design using low-fidelity prototypes. The resulting e-Channel system lets users create content channels—logical containers—for sharing on each other's displays. The key design insight was to separate the roles of content providers and display owners.

Content providers generate content (images, videos, webpages, live video streams) and organize it in channels. Content providers have full ownership of their channels and can add and remove content and set the date and times it is available. Content providers do not necessarily also control displays, and they have no say about where or when their content is displayed. Instead, channels can be shared and their content scheduled to a display after its owner subscribes the display to the channel.

Display owners control one or more physical displays, typically in a local physical space for which they have oversight. For example, college officers are normally display owners for displays located within their college. Display owners control their displays' channel subscriptions. A display can be subscribed to zero or more channels. Owners can also set the time at which their displays turn on and off.

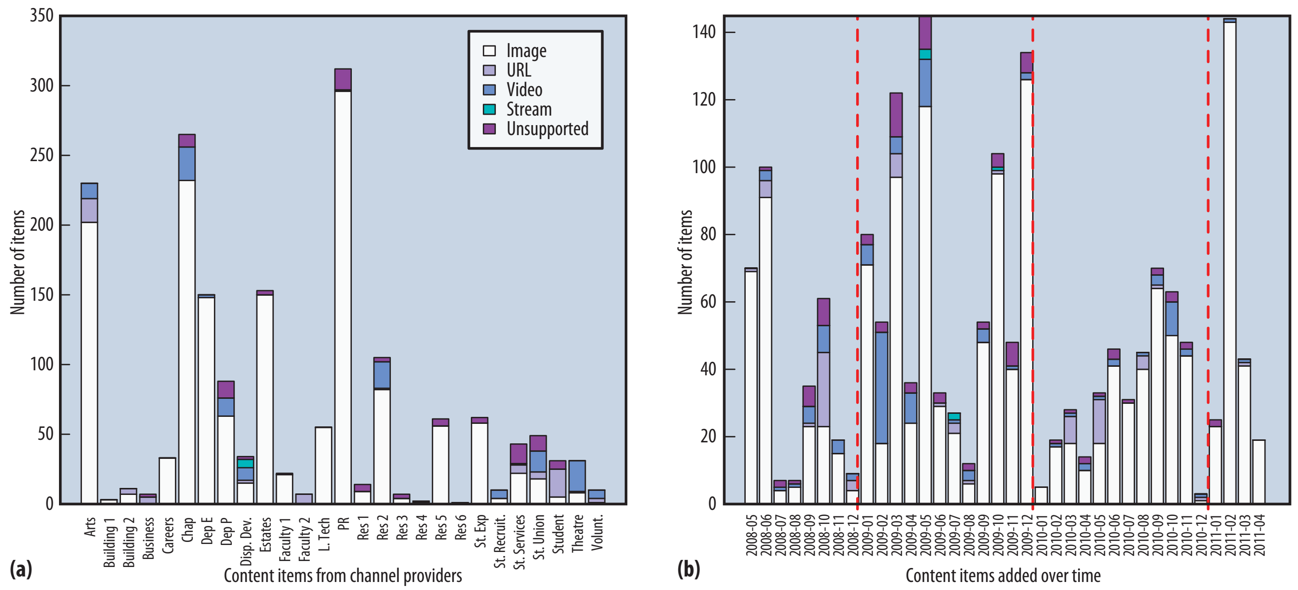
Each channel receives an associated network file share so that content can be managed conveniently via drag-and-drop. Content providers limit the availability of their channels using the e-Channel Web interface. A similar interface gives display owners control over their subscriptions. Users who are both content providers and display owners can create private channels available only for their displays.

The system produces a schedule for each display based on the combined set of content items from all its subscriptions. We weight all channels equally, although we give higher priority to newer or less frequently played content, and specific types of content such as emergency alerts and interactive applications.

Usage

We have found the e-Channel system, in continuous use since May 2008, to be highly effective in our university environment, which has largely trusted stakeholders. The system lets owners adapt their display's use to their particular context and needs.

Although owners cannot "see into" channels at the subscription point, we have observed from the usage traces of e-Channels that they do trust one another as content providers—that is, owners actively subscribe to each other's channels. Similarly, content providers are responsible for ensuring that content in their channels is appropriate for the displays' audience and that, for example, copyright ma-



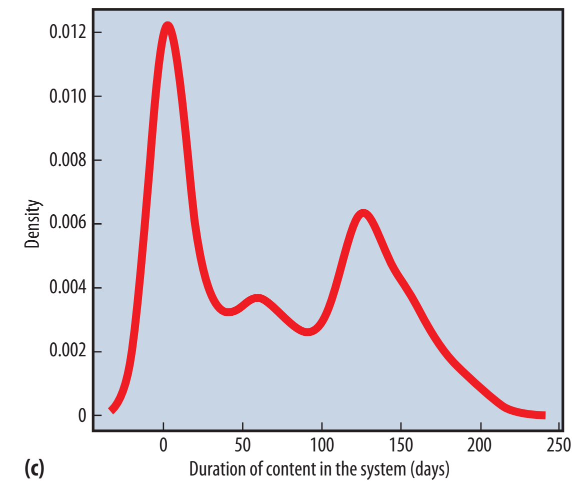


Figure 2. Analysis of content type, media, and lifetime. (a) Number of content items of each type by user group. (b) Timeline of number of content items added each day. (c) Content lifetime (probability density).

terial is not shown—there have been almost no instances of abuses of this trust, and no obvious copyright violations.

As Figure 2 shows, considerable variation exists between the different user groups and stakeholders in our system. The frequency of content generation varies dramatically—for many users, the system must be optimized for occasional use.[4] The split between shared and non-shared (private) channels is almost exactly 50:50; however, there are extremes: some users create primarily shared content, whereas others focus on producing private content for their own display.

Content creators use channels effectively as organizational tools for grouping content. The incentive to keep reusing public channels is that in doing so, display owners are not burdened with the need to continue modifying their subscriptions. Some users have exploited this reuse, going directly to the file share for their channel and manipulating their content files without going via the Web interface at all.

By far the most common type of content is images (83 percent), followed by video (7 percent). Anecdotally, it is easier to produce images given the expertise, time, and tool chains available. The content lifetime exists for about three months, loosely following the university calendar, but news and event announcements are in the system for considerably shorter periods.

Several user groups have developed a sophisticated understanding of how the system works and can subvert it to a certain extent: multiple copies of content means it will appear more often; creating long videos of static content essentially tricks the scheduler into presenting it for longer periods—and this is arguably simpler for users and developers than offering more complex controls over content play-out. This indicates that users will appropriate the system and bend it to their needs, providing it is sufficiently intuitive and flexible. Such abuses, while ostensibly a plus in situations where they have been used benignly to raise visibility of important content (as in our case), could be misused in other settings.

We have had only two incidents where we have been asked to act to remove content. In one instance, due to "decontextualization," a video with sound was shown during an exam (the providers could not have known how their channel was to be used, of course). We are more careful now about enabling audio in certain locations. In the second instance, resulting from the display's situated nature, an administrator was concerned that the audience would mistakenly perceive that the college was endorsing the antireligious viewpoint of one content item.

Both examples underscore the importance of being able to trace a content item to its source so that we can respond to requests from stakeholders to remove inappropriate content.

SUPPORTING DEVELOPERS

Early in our work, we wanted to offer a software system that let researchers and content authors create sophisticated applications for presentation on e-Campus. Inspired by recent work on tuple spaces, we conceived of an environment in which content creators could simply inject content together with a series of constraints into the system so that the content would be scheduled and displayed accordingly. In this way, we believed we could free authors from concerning themselves with the various video and audio sources/sinks and switching operations that might be required to load their particular content onto the displays.

In earlier explorations with a commercial signage solution, we had already found that traditional timeline-based programming (such as a TV schedule) was both too inflexible and too burdensome for our users, so we already knew that designing the scheduling constraint system would be a significant effort.

After more than a year, our attempts to automate creation of a content program for the enlarging network of displays continued to be foiled by the need to meet seemingly contradictory requirements, including the following:

- deterministic placement of content sequences on a display,
- control over the time and order in which the sequences are placed,
- support for the rapid introduction of interactive content (triggered, for example, by user presence or interaction),
- synchronized presentation of certain content across more than one display, and
- control over the transition between content items to achieve aesthetically pleasing and error-free presentation to viewers.

As is so often the case, failure to solve this problem led to a new and unexpected insight. Our innovation was not to create a single scheduler to meet all these requirements, but rather to separate the system's core functionality into an installation-independent computational model and an associated API that let us create special-purpose schedulers. For example, a round-robin scheduler might walk through an ordered content sequence, whereas an interaction scheduler might trigger a piece of content when the system detected a certain user in front of a display.

At some level, however, we conceded defeat—accepting that we were unlikely to ever create a single all-encompassing scheduler and recognizing that many schedulers would need to exist for a range of different purposes.

The computational model that emerged is an abstract form of a typical hardware deployment. It consists of a small number of conceptual entities: displays, applications, schedulers, and handlers. A logical display might represent a physical display, but could also be a specific region on a screen or a particular frame buffer.

In most cases, applications are actually wrappers around content renderers for image, movie, or webpage media types. Schedulers are the back-end logic of an application, running on an application server. Handlers resolve conflicts for physical display resources between logical displays and deal with any hardware-specific issues for a display such as monitoring the maximum on-time for a projector. We write new schedulers whenever a requirement is introduced for which the existing set of schedulers do not meet its needs.

The scheduling API consists of four conceptually simple operations that are applied to displays: CreateApplication, ChangeState, Transition, and TerminateApplication. These operations are surprisingly powerful: arguments can refer to multiple displays or groups of applications. Operation sequences can be associated with a transactional block that can fail atomically—enabling operations that involve multiple displays or multiple content items that must be available simultaneously to either succeed or fail without affecting the displays' visual state.

This Python code snippet illustrates play-out synchronization of two pieces of content across two displays:

```
try:
    gid = api.MakeGroupId()
    t = transaction( api, None )

    # Create renderers
    (worked, pid) = t.CreateApplication('display-1',
       "http://e-content/~demo/cycling1.mpg", gid)
    (worked, pid) = t.CreateApplication('display-2',
       "http://e-content/~demo/cycling2.mpg", gid)

    # Cause renderers to prefetch content (note
        use of group id)
    t.ChangeState(gid, APPLICATION_STATE_PREPARED)

    # Make content visible
    t.Transition(DISPLAY_ID_ALL, gid,
        APPLICATION_STATE_VISIBLE)
    t.commit()

except transaction_aborted, msg:
    print "Cant display cycling video", msg
```

The display components arbitrate requests using a simple scheme based on pre-emptable locks (an application has a display resource providing it is available or if the requesting application has higher priority than the application already holding the resource).

We implemented the computational model as a set of Python processes that communicate via a shared publish-subscribe event channel. This approach has several advantages, including allowing inspection of the system at runtime to determine its status or liveness and providing support for post hoc analysis after failures to determine the cause. Events can also be injected or scripted to manually orchestrate and extend the system.

The event channel, however, also proved to be the Achilles' heel of the entire e-Campus system: communication failure between the displays and the event channel was the most common cause of lack of e-Campus services.

We have used the low-level API to create schedulers for specific purposes, including a primary scheduler for handling the placement of day-to-day digital signage content, and various interaction schedulers (games, digital posters, and so on) that are triggered by user interaction, including SMS messages or Bluetooth sightings.

Although this API gives e-Campus developers complete flexibility, it also became clear that we needed a high-level API such as Ajax that was simple to use from within Web-based applications. We created an HTTP-based API for this purpose that enabled applications to create a request to display a playlist of content.

A fairly powerful set of optional display constraints can be set including the time, duration, number, and gap between repetitions, displays, or presence of particular users, detected using a Bluetooth media access control (MAC) address. A scheduler process evaluates these constraints and uses the low-level API to display the content when the appropriate conditions are met. We have found the main operations CreatePlaylist, ClonePlaylist, UpdatePlaylist, and CreateRequest to be sufficient. These abstractions (although not the API itself) underpin the e-Channel system.

APPLICATIONS

Over the years, we have formulated various theories about the kinds of applications that would stimulate engagement between displays and users. It would be fair to say that none of these applications have reached anything approaching regular or continued use. Here, we explore several possible causes for this lack of engagement.

Interactive applications and personalized content

We have created a wide range of applications that support interaction or provide personalized content, including the personalized pages displayed years ago as part of FLUMP. Within e-Campus, we initially explored the use of SMS messages to support interaction with a map tailored to a display's location. Users could text to an advertised phone number to trigger the map and highlight a route to their destination. A downloadable J2ME application offered lower-latency interaction via Wi-Fi for compatible phones.

Neither method had any lasting impact: the unpredictable latency, potential cost, and effort of sending commands via SMS posed significant usability problems. The return on the effort invested by the user installing the application was not high enough since the application was narrowly focused on the map, rather than on wider interaction possibilities.

To overcome the limitations of SMS and phone applications, we investigated Bluetooth as a possible alternative.[5] The main idea was to exploit Bluetooth-enabled mobile phones, using their device name as a method for issuing commands to tailor the content of nearby displays. The advantages were that this technique incurs no cost, there is no need to install an application, and delays are bounded if not entirely consistent. We supported commands to access Flickr photos, YouTube videos, Google keyword searches, and webpages via tinyurl.com short links.

Students at Lancaster developed one of the most imaginative pieces of content: *Capture the Flag*, a simple mixed-reality game.

Although they found the system easy to use, and initial concerns about privacy or users' reluctance to change their Bluetooth names appeared to be unfounded, users simply did not find the applications compelling. Despite an extensive marketing push during a campus open day with many thousands of visitors, we managed to attract only a handful of users.

For various reasons, including power conservation, many phones do not turn Bluetooth on or keep it on by default—in some cases such as the Apple iPhone, the name cannot be changed without docking the device, substantially undermining the technique for these users. We have also used simple Bluetooth scanning to trigger content, but the time it takes to complete a Bluetooth scan makes this approach far from ideal.

Students at Lancaster developed one of the most imaginative pieces of content: *Capture the Flag*, a simple mixed-reality game. Based on the popular game of the same name, players had to capture an enemy stronghold (particular displays) by grouping together in front of it until their Bluetooth devices were detected. Naturally, the enemy players competed to do the same thing. The winners were the team that captured the most displays.

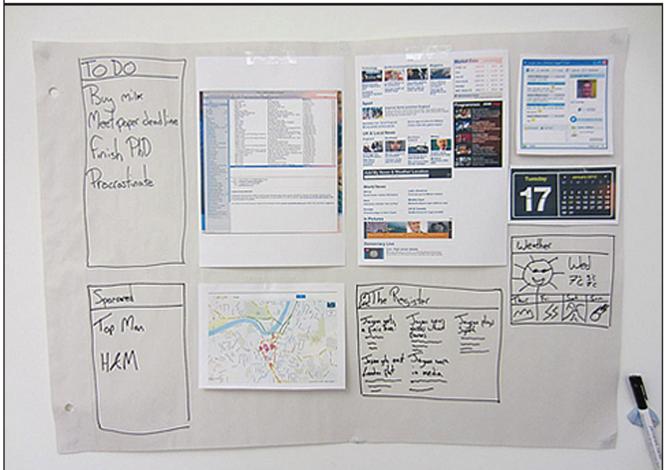


Figure 3. Sample screen from a participant in a study that explored the type of information university students might want displayed publicly.

The game was surprisingly exciting to play, although it caused considerable consternation to others as the players ran around campus. This is one of the few uses that has exploited the physical separation and situated nature of the displays, although we feel confident that other applications in this class are feasible.

Similarly, our most exciting context-triggered content, not least in terms of our enthusiasm, has been through collaborations with artists. This has led to the display of well-orchestrated sequences of content such as segments of poetry and video clips that are triggered in various places and times as the user is detected at each display.

Whereas with the game, delays in Bluetooth scanning can become part of the gameplay excitement, it is not well suited for detecting when users stand in front of or gaze at a display—in part due to the mismatch between gaze, line of sight, and the Bluetooth radio range.

Although contemporary tracking techniques such as Wi-Fi positioning might help to improve our ability to detect when users are nearby, we have not yet found an interactive application that has lasting impact.

Social media's role

We have made various attempts to engage students and lower the barrier of entry for posting content by linking into social media systems. Inspired by paper posters displayed around campus, we created a Facebook application allowing students to post digital posters with customized messages to interest groups and specific friends whose phone's Bluetooth MAC address is known.

Although some of these systems have not been widely released because of potential privacy concerns, we have explored applications in this vein in focus groups and surveys about the kinds of applications users would like to see on the displays.

Whereas the poster application divides opinion but could have some traction, social media feeds—perhaps due to their sometimes intimate nature, or the way they are often consumed on personal devices—are not considered appropriate information sources for public display.

The changing face of display appropriation

The ability to appropriate public displays to enhance the capabilities of resource-limited mobile devices has remained a consistent and compelling application scenario for our research. However, the form that this appropriation might take has changed significantly. When we began our research in 1996, the Web was in its infancy and appropriation tended to focus on concepts such as the personal server,[6] in which a user could take control of an entire display to compensate for the lack of screen real estate.

With the Web's emergence, the emphasis shifted to appropriating displays to surf the Web: the assumption was that mobile devices would not be sufficiently capable of doing this. The emergence of smartphones with large multitouch displays has made this assumption invalid, and display appropriation has now entered into a new era in which researchers are focusing on appropriation for information presentation or personalization. The idea of appropriation to supplement mobile devices continues to be articulated as part of the cloudlet vision, and some compelling scenarios are emerging.[7]

CHANGING REQUIREMENTS

We might expect that the applications users want to run on pervasive public displays would have changed radically within the past 16 years. Although we have seen the emergence of entirely new classes of applications during that time that have transformed desktop and mobile usage patterns, the applications conceived for public displays have shown remarkable resilience to change.

Our first-ever attempt at a public display system in 1996 showed a carousel of news and departmental events that could be interrupted by personalized content when the system detected a user's presence. This personalized content usually consisted of a tailored news and sports feed, local weather updates, and information about the user's e-mail (typically the new message count).

Earlier this year, we ran a series of studies in which we explored with students what sort of information they might want to show on a public display. We disseminated an online survey, and a total of 68 respondents provided information on the likelihood that they would use a set of predetermined applications including Facebook, Twitter, Flickr, news, e-mail, gaming, a clock, and maps. We found a strong preference for traditional signage content such as news, maps, and, perhaps surprisingly, a clock.

To gain further insights, we conducted two focus groups in which participants were given the task of assembling their own personal display desktops using paper resources.

As Figure 3 shows, we provided a public display of a paper and a selection of printed screenshots for participants to use in creating their own desktop. These screenshots included news, social networks, maps; common widgets such as clocks, calendars, weather, sticky notes; and other typical applications: e-mail, instant messaging, and games. We encouraged participants to use scrap paper and colored pens to create additional applications as needed. Participants described their desktop and the motivation for their choices to the group.

During these sessions, we found that participants openly discussed issues including privacy, the content they would place on the display specifically to share, the effect of display location, and the desire for personalization.

Finally, we asked participants to consider what they would want to show if more than one user was within display range. Once again, all participants included a significant amount of traditional signage content such as news feeds and weather reports. Most participants wanted to see their own, rather than generic, news feeds—a strong requirement for personalization. Reinforcing our findings from the survey, many participants created sketches that included clocks and calendars.

Anecdotally, we noticed that staff and students often no longer wear watches, preferring to use their mobile phone for telling time; perhaps their enthusiasm for clocks on public displays reflects the additional effort required to check the time on a phone rather than a watch.

Apparently, despite all the radical changes during the past 16 years in how we interact with applications and information, potential display users are drawn to the same set of applications that were conceived more than a decade ago.

Conducting this type of longitudinal work in the public domain presents challenges.

In our experience, recognizing the public nature of the work and actively seeking to involve the various stakeholders is crucial to the success of any long-term deployment. Equally important is the effort to manage expectations about the system, especially early on: in situ testing can be confused with a live system, and periods when the system is deliberately off can be mistaken for system failure.

A further consequence of the public nature of this work is that researchers in this domain must prepare for public scrutiny—there will be curiosity about the system, and researchers might be held accountable to funders, media, or the public.

Despite these challenges, constructing and maintaining a large-scale, long-lived research deployment that remains in everyday use for many years offers unique insights into the domain that simply cannot be obtained any other way.

These reflections on our research are intended to prove valuable to those considering research or practical deployments of public displays. C

Acknowledgments

We thank the European Union Seventh Framework Programme (FP7/2007-2013), grant ref. 244011; Lancaster University for their support; and Oliver Storz for his outstanding contribution to the e-Campus infrastructure.

References

1. J. Finney et al., "FLUMP: The Flexible Ubiquitous Monitor Project," *Proc. 3rd CaberNet Radicals Workshop*, 1996; http://tinyurl.com/flump1996.
2. K. Cheverst et al., "Exploring Situated Interaction with Ubiquitous Office Door Displays," *Proc. Conf. Computer-Supported Cooperative Work, Public Community and Situated Displays Workshop*, ACM, 2002.
3. O. Storz, A. Friday, and N. Davies, "Supporting Content Scheduling on Situated Public Displays," *Computers & Graphics*, vol. 30, no. 5, 2006, pp. 681-691.
4. S. Clinch et al., "Reflections on the Long-Term Use of an Experimental Digital Signage System," *Proc. 13th Int'l Conf. Ubiquitous Computing* (UbiComp 11), ACM, 2011, pp. 133-142.
5. N. Davies et al., "Using Bluetooth Device Names to Support Interaction in Smart Environments," *Proc. 7th Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys 09), ACM, 2009, pp. 151-164.
6. R. Want et al., "The Personal Server: Changing the Way We Think about Ubiquitous Computing," *Proc. 4th Int'l Conf. Ubiquitous Computing* (UbiComp 02), LNCS 2498, Springer, 2002, pp. 223-230.
7. M. Satyanarayanan, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Internet Computing*, vol. 8, no. 4, 2009, pp. 14-23.

Adrian Friday *is a senior lecturer in the School of Computing and Communications at Lancaster University, UK. His research interests include the challenges of ubiquitous computing deployment, open public display networks, and the application of ubiquitous computing to mitigating climate change. Friday received a PhD in distributed systems support for mobile computing from Lancaster University. Contact him at a.friday@lancaster.ac.uk.*

Nigel Davies *is a professor in the School of Computing and Communications at Lancaster University. His research focuses on mobile and ubiquitous computing. Davies received a PhD in computer science from Lancaster University. He is a member of ACM and IEEE. Contact him at nigel@comp.lancs.ac.uk.*

Christos Efstratiou *is a senior research associate in the Computer Laboratory at the University of Cambridge, UK. His research interests include challenges in deploying sensor networks in smart buildings and industrial environments, open public display networks, and mobile and ubiquitous computing applications. Christos received a PhD in context-aware and adaptive mobile systems from Lancaster University. He is a member of ACM and IEEE. Contact him at christos.efstratiou@cl.cam.ac.uk.*