

Key Challenges in Application and Content Scheduling for Open Pervasive Display Networks

Ivan Elhart^{*}, Marc Langheinrich^{*}, Nigel Davies[†], and Rui José[‡]

^{*}Faculty of Informatics
University of Lugano (USI)
Lugano, Switzerland

{ivan.elhart, marc.langheinrich}@usi.ch

[†]School of Computing and Communications,
Lancaster University
Lancaster, UK

nigel@comp.lancs.ac.uk

[‡]Algoritmi Research Centre
University of Minho
Guimarães, Portugal

rui@dsi.uminho.pt

Abstract—Today’s digital signage systems typically show content that has been scheduled well in advance by their respective “owners”, i.e., companies or individuals who paid for and/or operate the public display. However, with the shift to open display networks that can obtain content from many sources and the corresponding advances in interaction and sensing technologies, the scheduling requirements in this domain are set to change radically. For example, we envision that displays in our environment will soon be able to adapt to their surroundings and allow viewers to appropriate them by actively selecting and/or contributing content. Such levels of interactivity and context-awareness will require new approaches to content scheduling. In this paper we discuss the challenges faced in developing new forms of application and content scheduling for Open Pervasive Display Networks.

Keywords - public displays; scheduling

I. INTRODUCTION

In order to increase the overall utility of public displays, we envision the creation of large-scale networks of displays and associated sensors that are open to applications and content from a wide range of contributors [1].

Opening public displays to many contributors creates new challenges that involve the control and selection of content. Our main research question is: how can we open today’s closed and isolated public displays to third party applications and user-contributed content, but still keep their “owners” in control? While there are different social and economic issues surrounding this vision (see [1] for a discussion), we have started looking into the various technical challenges that arise from opening up displays to multiple dynamically scheduled applications. How can individual applications dynamically schedule new content on a public display? How can a public display adapt to environmental parameters and/or the individual preferences of passers-by? And how can we allow viewers to explicitly show their own content, given an existing schedule?

We believe that creating appropriate scheduling mechanisms and policies for open pervasive display networks presents fundamental new challenges for the research community. In this paper we describe some of these challenges and begin to present a conceptual framework that may help to structure future contributions in this area.

II. SCHEDULING IN OPEN PERVASIVE DISPLAY NETWORKS

In previous work, we identified four key factors that influence content selection on public displays: display owner preferences, display viewer preferences and presence, content availability, and a display’s contextual situation [2]. Given an open network of displays, these factors can significantly complicate the deceptively simple-looking task of showing content on a display. Examples of scheduling requirements that might arise from such settings could be:

- show a piece of content at a specific time, e.g. show the news at 6pm.
- show a piece of content just before one other piece of content, e.g. always show the weather forecast before the news.
- show a piece of content a specific number of times within a given time frame, e.g. show a piece of content 20 times a day to meet an advertising commitment.
- ensure two or more content items are separated by a given time interval, e.g. ensure adverts from rival companies are not shown too close together.
- show content across multiple displays, e.g. showing a multimedia presentation that spans across several displays and requires all the displays to be available at the same time.
- show content as a result of contextual triggers, e.g. show content when a specific viewer (or class of viewer) is present – or indeed the inverse which could be used, for example, to ensure content displayed is age-appropriate.
- show content in response to sensor events, e.g. as a result of a user scanning a product. This could be considered a further example of a contextual trigger.
- pre-empt content on the screen in favor of higher-priority content, e.g. to display an emergency notice.
- ensure a specific viewer (or class of viewers) sees a piece of content a set number of times within a given time frame.

The above list is not intended to be comprehensive – rather it provides a series of examples that illustrate the scope and complexity of the challenge and also how scheduling requirements may span across different abstraction levels and control contexts.

A number of projects have already started to address this vision of open display networks, such as e-Campus [3], Instant Places [4], and UBI-hotspots [5]. While all three systems support different application and content scheduling requirements, only e-Campus provides flexible ways for developing independent constraint-based schedulers. However, while e-Campus enables developers to construct arbitrary schedulers it says nothing about the actual behavior of such schedulers nor does it provide support for dividing screen real estate between applications.

It is the challenge of developing appropriate scheduling mechanisms and policies that we focus on for the remainder of this paper. In particular, we consider the key factors and scheduling requirements in three broad areas: *long-term app scheduling*, *in-app content scheduling*, and *real-time app composition*. These three areas are described in the subsections below and illustrated in Figure 1 below.

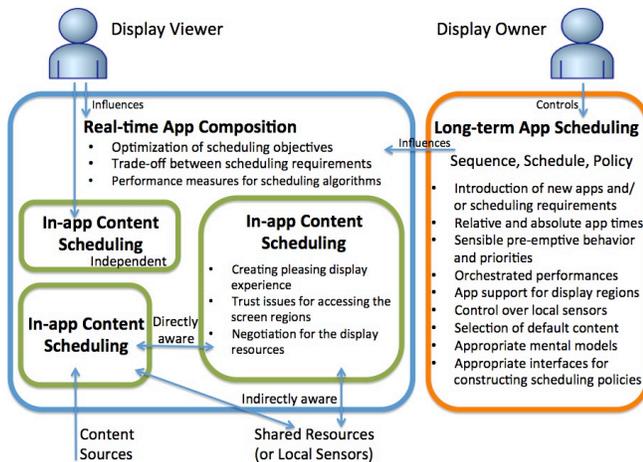


Figure 1: Three areas that influence scheduling on open pervasive displays: long-term app scheduling, in-app content scheduling, and real-time app composition.

A. Long-term App Scheduling

The long-term app scheduling relates to the overall control of public display resources and allows display owners to plan global scheduling. It represents an important point of control over the level of openness of a display network between two extremes: tightly controlled traditional digital signage and totally open display installations with no content control.

General scheduling theory often differentiates between a *sequence*, a *schedule*, or a *scheduling policy* [6]. In the context of public displays, a sequence would be a predefined and well-orchestrated list of content items with precisely defined timing, presentation styles, and content transitions. A schedule would be a similar list of content items, yet allowing for the pre-emption of content by other items that get added at later points in time. In the dynamic and real-time settings that we envision, the concept of a scheduling policy is much better suited to describe display behavior. A scheduling policy usually prescribes rules,

describes appropriate actions, and allows for the introduction of additional parameters that can influence the real-time composition. A scheduling policy can be seen as an extension of a schedule, where pre-emption, priorities, application requests, and other scheduling parameters can continuously change over time.

Note that while traditional signage systems tend to describe scheduling in terms of media items, we focus on the more general concept of scheduling *applications* and their *tasks*. A simple static playlist, e.g., a set of slide-show presentations, would thus be modeled by scheduling a single application – a slide-show viewer – with a set of tasks – the individual content items.

Depending on the amount of freedom that the real-time app composition (cf. section C below) can have in making the final scheduling decisions, we can group an open pervasive display’s scheduling policy into one of four broad classes [6]: (1) *Non-pre-emptive Static List Policies*, where all applications and tasks are known in advance and their presentation times are ordered according to a priority list – similar to today’s digital signage; (2) *Pre-emptive Static List Policies*, where the individual items are well-known in advance but their display can be interrupted, e.g., to warn of an incoming train in a digital signage system at a train track; (3) *Non-pre-emptive Dynamic Policies*, where the decision of which application to show next is made only when the current application – which cannot be interrupted – finishes its presentation; and (4) *Pre-emptive Dynamic Policies*, where the decision to show an application can be made at any point in time, even when an application is currently still running. While the last class presents the most flexible scheduling policy, it also is the most difficult to predict, making it hard for display owners to understand what type of content is shown when on their display.

Supporting such a dynamic scheduling of concurrent applications leads to a number of key challenges:

- *Real-time introduction of new applications and/or scheduling requirements.* Dynamic application scheduling allows apps to independently request display resources without explicitly being scheduled by display owners. This not only requires a powerful real-time application composition engine (cf., section C below) but also adequate levels of control for display owners, such as allowing only applications signed by a third-party company, or based on application ratings.
- *Controlling relative and absolute application times.* When applications can request display resources and be scheduled based on local sensor inputs, it becomes difficult to predict their actual playtimes. Owners thus need controls to limit the airtime of an application, either in absolute terms (e.g., number of minutes, or time of day) or relatively to other applications.
- *Sensible pre-emptive behavior and priorities.* While emergency applications should be able to pre-empt any other application on the screen, care must be taken when simply defining priorities for other applications. If user-

initiated, on-demand apps by default come with a lower priority than owner-initiated scheduled content, an interactive app (e.g., a game) might get suddenly stopped by a non-critical yet higher level priority app.

- *Supporting orchestrated performances.* It should be possible to specify and support orchestrated performance, i.e., a precisely defined sequence of applications such as “show the weather application immediately after the news application.” If the display real-estate can be divided into independent regions, orchestration would entail the coordination among different apps active in different parts of the screen, e.g., a ticker-tape app running concurrently to a main application. Pre-emptive dynamic policies might not only make such coordination difficult to predict and/or guarantee, but also complex to model.
- *Application support for display regions.* Applications accessing screen regions will have to adapt their presentation size and resolution accordingly, which might increase their complexity and thus costs.
- *Control over local sensors.* It should be possible to control which local sensors can influence application scheduling. Individual sensor values or aggregate values from multiple sensors can also influence some of the parameters of application scheduling over time, such as priorities.
- *Selection of default content.* A public display should typically show content all the time. In situations when there is no application requesting airtime, when there are gaps in applications scheduling, or when off-line operation has to be supported, it is desirable to have default content for presentation. Default content can be simple as a screensaver that shows static images.
- *Appropriate mental models for scheduling.* Timelines are an established metaphor for communicating schedules in traditional digital signage. E-Campus uses a “bucket” metaphor for communicating the system’s scheduling behavior to users. Highly dynamic policy-based scheduling needs equally powerful mental models to enable display owners to use the system.
- *Appropriate scheduling interfaces for constructing scheduling policies.* It is unlikely that display owners will have the time and energy to construct complicated scheduling policies. Long-term app scheduling must offer powerful visualization tools, such as a preview of applications and content that will appear on the screens, and how the change of one scheduling parameter would influence the overall display behavior.

B. In-app Content Scheduling

The long-term app scheduling influences which, when, and how public display applications will be shown on one or more displays. Once an application is ready and scheduled for presentation, the *in-app content scheduling* is concerned with when and how individual content items within the application are shown on the screen. This encapsulates

within the application much of the complexity that is normally associated with rich animation of visual content.

Applications can present content on one or more displays and within different screen regions (different sizes and resolutions). Presentation of an application content on multiple displays is mainly concerned with synchronization of multiple instances of the application or multiple application clients. However, when a display is divided in different screen regions, multiple concurrent applications can run and show content at the same time. Public display applications that run concurrently and share the screen area may or may not interact with each other. Based on the level of coordination there can be three types of interaction:

- *Applications unaware of each other.* They concurrently compete and negotiate for display resources and independently schedule content items for presentation.
- *Applications indirectly aware of each other.* They share access to a common resource such as a database or a local sensor input. They independently make decision to show content, but the shared resource may influence their decisions. This may result in showing similar and related content at different regions at the same time.
- *Applications directly aware of each other.* They directly communicate and cooperate in presenting content on the screens. One application can call other applications and pass parameters for visualization in the same or a nearby screen region.

Content within concurrently running applications can come from services (e.g. automatically generated based on local sensor inputs, global news, etc.) or can be viewer-contributed (e.g. tweets, images). Such content is usually not known in advance and can be hardly previewed before appearing on the screens. This creates the additional issue of content moderation. It is unlikely that display owners will spend time to moderate individual content items within each application before the presentation time. However, applications may provide different tools and mechanisms of content moderation that could filter inappropriate content.

Key challenges in in-app content scheduling include:

- How to *create aesthetically pleasing display experiences* when multiple applications with different styles can access different screens regions
- Applications might need to provide additional information to display owners to *gain their trust for accessing the screens* (e.g. description of content, content examples, screenshots, ratings by other display owners and viewers)
- How to support applications that can *negotiate for the display resources* (airtime, screen regions, access to local sensors)

C. Real-time App Composition

The Real-time app composition is responsible for making the final and fine-grained decision which application or a set of applications to show on one or more display regions and on one or more public displays. It

makes the decision which application to show whenever one of the previous applications finishes its presentation, when one of the screen regions become available, when there is a higher priority task, when there is a local sensor input, or when an application requests the access to a screen region. The final decision is based on the parameters of the long-term app scheduling and a local scheduling algorithm.

The local scheduling algorithm can treat the collection of available applications as a single pool from which to select the next application [7]. The local scheduling algorithm can be based on one of the following algorithms:

- *Priority based*, where each application is assigned a priority and the algorithm always chooses an application of the highest priority. However, if there is more than one application of the same priority, the algorithm may use an additional priority policy to make the decision. A downside of using this type of algorithms is that they may lead to application starvation, a situation when low-priority applications are never shown because of a constant presence of high-priority applications.
- *First-come-first-served*, where applications form a queue of ready applications waiting for the airtime. When the current application finishes its presentation, the application that has been in the queue the longest is selected for presentation. A downside of this type of algorithms is that a time critical application may need to wait for all low-priority applications that are longer in the queue to finish.
- *Round robin (or time slicing)*, where the algorithm makes the decision which application to show at periodic intervals. At the end of the interval, the algorithm pre-empts currently running application and selects the next application according to additional scheduling criteria. The duration of the time interval is the main issue for this type of algorithms. Since interactive applications are of different duration, a short time interval may often interrupt the user interaction.

Additional more complex scheduling algorithms may be constructed based on dynamic scheduling policies. The goal of scheduling algorithms is to optimize the display behavior based on one or more scheduling parameters described in a scheduling policy. The parameters might be interdependent and the optimization of the display behavior may involve compromising among competing requirements. Based on the intended use of a public display, a scheduling policy may include the relative weights of the various scheduling parameters. Also, the optimization will need to balance the following general characteristics: (1) *Predictability*, i.e., the extent to which a public display behaves in similar way in different situation given the same or similar set of requirements; (2) *Responsiveness*, i.e., the amount of time between an application request until the application is shown on the screen; (3) *Controllability*, i.e., offering

owners fine-grained control over application scheduling and diverse scheduling parameters in real time at the cost of complexity; and (4) *Reliability*, i.e., the ability of a public display to handle degradation of application performances, e.g., creating a subtle transition between applications and the default content when applications become unavailable.

Key challenges in real-time app composition include:

- Optimization of one or more *scheduling objectives*
- Managing the *trade-off* between interdependent and competing scheduling requirements and objectives
- Performance and comparison *measures* for different scheduling rules and algorithms

III. CONCLUSION

We believe that future networked public displays will be open to diverse applications and content sources. Such openness to applications would enable individual public displays to allow display viewers to appropriate displays, run favorite applications, and contribute their content; and still keep public display owners in the overall control of their displays. In this paper we have described some of the key challenges in supporting concurrent application and content scheduling for open display networks. We believe this will help the community understand and conceptualize scheduling mechanisms and policies for supporting concurrent applications and personalization by both display owners and display viewers.

ACKNOWLEDGMENT

The authors acknowledge the financial support of the Future and Emerging Technologies (FET) programme within the 7th Framework Programme for Research of the European Commission, under FET-Open grant number 244011.

REFERENCES

- [1] Davies, N., Langheinrich, M., José, R., And Schmidt, A. 2012. Open Display Networks: A Communications Medium for the 21st Century. *IEEE Computer*. 45, 5 (Special Issue on Interactive Digital Signage), 58 – 64.
- [2] Elhart, I. 2012. Control and Selection of Content on a Network of Public Displays. PhD Colloq., Pervasive 2012. Newcastle, UK, 2012.
- [3] Storz, O., Friday, A., & Davies, N. (2006). Supporting content scheduling on situated public displays. *Computers & Graphics*, 30(5), pp. 681-691, 2006.
- [4] R. José et al., “Beyond Interaction: Tools and Practices for Situated Publication in Display Networks,” *Proc. Int’l Symp. Wireless and Pervasive Computing (ISWPC 12)*, ACM, 2012
- [5] Linden, T., Heikkinen, T., Ojala, T., Kukka, H., and Jurmu, M. 2010. “Web-based framework for spatiotemporal screen real estate management of interactive public displays,” In Proceedings of the 19th international conference on World wide web - WWW’10, New York, New York, USA: ACM Press, pp. 1277.
- [6] Pinedo M. L. *Scheduling: Theory, Algorithms, and Systems*, Third edition, Springer, 2008.
- [7] Stallings, W. *Operating Systems: Internals and Design Principles*, Sixth edition, Prentice Hall, 2009.