MODS: Modularly Operated Digital Signage

Rolf Egert
Technische Universität Darmstadt
Darmstadt, Germany
egert@tk.tu-darmstadt.de

Tim Grube
Technische Universität Darmstadt
Darmstadt, Germany
grube@tk.tu-darmstadt.de

Max Mühlhäuser
Technische Universität Darmstadt
Darmstadt, Germany
max@tk.tu-darmstadt.de

Abstract

Signage is transitioning from static analogue signs towards Digital Signage (DS), which introduces a variety of benefits. Among those are remote-maintenance, supporting dynamic content like videos and animations, and the simplification of updating content. However, DS solutions, despite being ubiquitous, are often tailored to specific use-cases, which limits their re-usability and updateability in case of severe changes to their environment. For instance, digital door signs for office spaces may become unusable if the office space is reorganized as storage or seminar rooms. Coping with such changes may result in additional costs since new DS solutions need to be purchased or in-depth changes to the software of the currently deployed DS solution are required. To address these problems we propose the Modularly Operated Digital Signage (MODS) framework, facilitating the dynamic changing of DS functionality in a modular fashion. We present the frameworks modular concept and describe its individual components. Subsequently, we briefly elaborate on the properties of the currently implemented modules. Additionally, we discuss the conducted pre-study to receive a first indicator for the usability of the framework.

Author Keywords

Digital Signage; User Interaction; Modularity

ACM Classification Keywords
H.4.m [Information systems applications:]: Miscellaneous

Introduction
Using signs for displaying, highlighting and conveying information is an established technique that is ubiquitous in our daily life [2]. These signs can range from small warning symbols, which are printed on gadgets and products, over door-signs used for identifying rooms in buildings, towards large scale advertisements. While some of those signs serve their purpose for a long time period (e.g., organization names, street signs), others may have shorter life spans, resulting in outdated or false displayed information. The consequences of the resulting misinformation can range from minor irritation to misunderstandings, which ultimately can be harmful if signs that are supposed to inform about dangerous aspects (e.g., dangerous locations) are not up-to-date. To address the problem of outdated information, signs need to be replaced regularly, which involves printing and the manual installation of the updated sign at all positions where it should be displayed. Nowadays, digitization aims at simplifying the task of displaying and updating information by using Digital Signage (DS) instead of static analogue ones. Alongside this transition towards DS come additional benefits. Among those are the capability of displaying dynamic content like animations or videos, the integration of sound, remote updateability and many more [3].

However, many solutions are still tailored towards a specific use-case, like showing a playlist of pictures and videos or presenting a web-page. One example of such an approach is Xogo [5], which allows displaying a predefined set of images and movies. However, the content is rather static and the user interaction is limited to change between the content locally available on the device. Other solutions that aim to increase the flexibility of interaction and functionality are based on Raspberry Pis. Among those solution are *piSignage* [1], which also facilitates playlist-based functionalities but allows for displaying additional content like web-pages or calendar events. Another approach is presented by the authors of [6], where Raspberry Pis are used to support the room management in university buildings and provide auxiliary information relevant for students. Other solutions are more powerful but are only available as commercial products. An example of such a system is [4]. The tool provides sophisticated functionalities, supporting remote-control via App, providing touch interaction for users and also enables querying information of different formats. The whole suit is controlled using a web interface. However, the tool is closed-source and, therefore, its extensibility is limited.

One of the main issues identified from related solutions is the limitation that DS is often tailored towards specific use-cases. This still limits the usability of many DS applications to the very domain they were developed for, which is problematic in case of severe changes happen to the environment. For instance, an office environment is rearranged as a combination of storage and lecture rooms, or additional use-cases were identified that are relevant for the current environment and are not covered by the functionality of the currently deployed software. If the previously installed DS does not support adequate functionality to fulfill the requirements, they need to be replaced or more drastic modifications have to be made to the software that is running on those devices. Consequently, this may lead to significant monetary investments, either for buying new products or for modifying the software and deploying it on all available devices.

In this context, we propose the Modularly Operated Digital Signage (MODS) framework, which addresses the missing dynamic adaption to multiple use-cases by organizing different functionality in a plugin-based fashion. This allows to easily change the appearance and functionality of the DS according to the current use case. Moreover, MODS introduces the concept of Interrupt Handlers as an event-based interaction mechanism independent of the current use-case. Additionally, we evaluated the usability of MODS in a pre-study.

The remaining paper is structured as follows. First, the general concept of MODS is explained and the currently implemented modules are briefly introduced. Second, the conducted pre-study assessing the usability of MODS is explained. Finally, the work is concluded and an outlook to future work is given.

MODS Concept

The purpose of MODS is twofold: first, it aims to provide the functionality that is generally supported by DS, like displaying dynamic content, enable user interaction and updating information; second, the framework addresses the problem of dynamically adapting to changing use-cases of the DS by enabling the exchange of functionality. Note that hereby two types of users have to be distinguished. One user is interacting with the DS to receive information and the second one (i.e., administrative user) is using the framework to display and update information. To facilitate the adaption of the DS towards multiple use-cases, the framework allows changing the running software to fit the desired use-case by loading the corresponding functionalities as modules. The general structure of the MODS framework is displayed as Fig. 1

The framework is separated into two main parts, namely *Sign Program*, which describes the software part that is running on the DS device and the *Update Program*, which runs on the users/administrators computer enabling remote

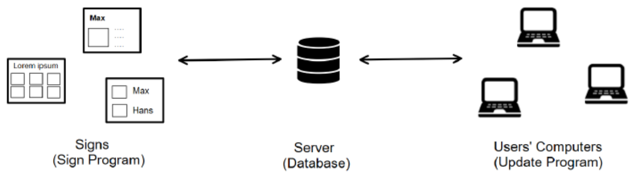


Figure 1: General three component concept of MODS

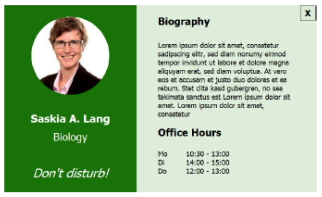
configuration and control. In the following, these two parts are introduced in more detail.

Sign Program
The *Sign Program* represents the core of the MODS framework and acts as the program that is running on the device used as a digital sign. This part of the framework allows realizing the end-user application (i.e., the visual and interactive part of DS which is perceived by the person using it) of the DS for a diverse set of use-cases in a plugin-based fashion. The available plugin types and their connections are described in the following in more detail.

Mode
A *mode* defines a set of use-cases the DS addresses and, therefore, provides a set of functionalities that are related to these use-cases. At each point in time, only one mode can be active on a DS. An exemplary mode can be a “booking mode” for seminar rooms that allows reserving dedicated time slots for using the room. While this mode is running, another mode cannot be executed simultaneously. In this scenario, a use-case represents the general application scenario where the DS is deployed, like an office situation or as an advertisement board. Each mode can be loaded in a plugin-based fashion, facilitating a simple way of changing the set of functionalities which is required by the DS. The visualized components of a mode are defined as so-

called *SignViewComponents*, which are part of different *SignViews* of a mode and are responsible for displaying arbitrary content. The SignViewComponents can receive interaction information from the user of the DS and forward it to the internal logic for further processing.

SignViews & SignViewComponents
The SignViews represent the different General User Interface (GUI) canvases or layers that are present in the currently active mode. Each mode can have multiple SignViews, which again consist of an arbitrary number of SignViewComponents representing the GUI parts of the currently displayed SignView. All SignViews are loaded when a mode becomes active, to reduce loading times between SignView changes. The SignViewComponents are responsible to update GUI information and to react to or propagate user interaction behavior.



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur

Saskia A. Lang
Biology
Don't disturb!

Office Hours

Mo	10:30 - 13:00
Di	14:00 - 15:00
Do	12:00 - 13:00

Figure 2: Detailed information view for individual people listed in the Office Mode.

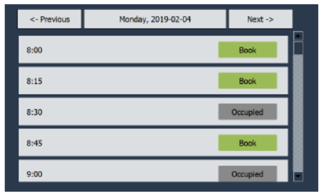


Figure 3: Conference View which enables the booking of individual time-slots at specific dates.

Interrupt Handler & Interrupt View
The two remaining types of plugins are the so-called *Interrupt Handlers* and *Interrupt Views*. These plugins differ from the previously introduced ones since they are independent of the currently active mode. Moreover, their main purpose is to actively interrupt the currently running mode and interfering with the currently displayed information. This is done by changing the active view to a so-called Interrupt View, which is displayed for a limited amount of time before it changes back to the previously displayed view. Therefore, Interrupt Handlers are continuously listening for specific events that trigger their execution.

Update Program
The *Update Program* is the second part of MODS and represents the administrative user side of the framework. It runs on devices that are used for maintaining and updating the DS content, like PCs or notebooks and provides functionality for visualizing DS content and modifying it. For this, the Update Program provides a general overview of all the signs that are currently registered. Selected signs can then be displayed in the Update Program as they would be running on the DS device. This can be done since many of the components that are already present in the Sign Program are re-used and extended with so-called *UpdateViews* and *UpdateViewComponents*. This re-use of components enables the Update Program to show and interact with the current content, as it is displayed by the real DS, and, additionally, the extensions provide modification and update capabilities.

For being able to provide both, authentic usability as provided by the DS device and the modification functionalities of the Update Program, it is necessary to distinguish interaction events w.r.t. two different aspects. First, the *context* in which a mode is executed, is used to define where the DS software is currently running. The two possible context options are either the genuine DS device, or the Update Program. The second aspect is the *action* that is conducted to interact with a component. For instance, for the normal interaction with a DS, touch inputs are interpreted as left clicks on specific displayed SignViewComponents. However, in the context of the UpdateProgram additional actions can be defined, which can be leveraged to trigger corresponding update functionalities. Therefore, if a left click is used for normal navigation purposes, a right click could be configured to trigger the implemented update functionality of the SignViewComponent (e.g., opening a picture upload dialogue). This does not have any effect if the software runs in the context of the DS but leads to executing update functionalities in the context of the UpdateProgram.

Office Mode
The *Office Mode* plugin is tailored to represent the use-case of a general office door sign. An exemplary application

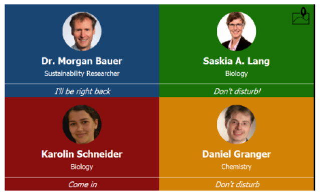


Figure 4: SignView representation

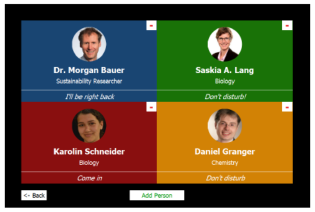


Figure 5: UpdateView representation

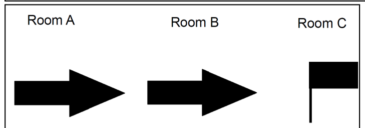


Figure 6: Exemplary symbols displayed after triggering the WayFinder Interrupt Handler. Arrows indicate the suggested walking direction for the users. The flag symbol indicates the goal destination.

for this mode is shown in Fig. 4. This mode provides an overview of the people present in the office, corresponding pictures and naming information. Additionally, a status is provided, which can be modified according to the preferences of the individual people. More detailed information about individual people can be shown by interacting with their profiles. This triggers the navigation towards a more detailed view, which may look as depicted in Fig. 2

Conference Mode
For simple room-booking and calendar functionalities tailored towards seminar and lecture rooms, the *Conference Mode* plugin can be used. Fig. 3 shows the current implementation that allows booking specific time-slots for various dates.

Message Interrupt Handler
The *Message Interrupt Handler* plugin is a prototypical way of communicating between a normal and an administrative user of a sign. In general, this interrupt handler plugin allows an administrative user to send messages to specific signs, which are then displayed for a configurable amount of time. Potential application cases include support-modes that allows administrators to react and directly inform users that retrieve information about problems using DSs.

WayFinder Interrupt Handler
A common problem in foreign environments like new cities or buildings is related to way-finding. The *WayFinder Interrupt Handler* is a plugin that allows a user to query the DS for directions towards, for instance, a specific room in a building, which is also equipped with a DS and selectable from a set of registered devices. The DS then displays one of the symbols shown in Fig. 6, indicating either a movement direction for the user using an arrow symbol or the destination location using the flag symbol. Additionally, the DS returns to its original view after a predefined amount of time and the next DS on the way towards the destined location is invoking the WayFinder Interruption handler to indicate the next steps for the user.

Pre-Study

For assessing the usability of both, the Sign Program and the Update Program a pre-study was conducted. The study encompassed eleven participants (i.e., scientific employees and students) from the domain of computer science. Therefore, a basic understanding of touch-controlled devices could be assumed for all participants. The participants had to fulfill tasks from three main categories: First, the participants had to conduct general navigation and interaction tasks using the DS device. Second, tasks that involved updating displayed information via the Update Program needed to be handled. Finally, the participants had to solve extended update tasks.

Setup
The setup of the evaluation environment consisted of three Raspberry Pi devices acting as DSs and three attached 7" touch screens for user interaction. Additionally, one notebook was set up, which was used to run the database and the Update Program part of the framework. The three Raspberry Pis were configured to represent three door signs in an office environment (i.e., the mode was set to Office Mode). The participants had to conduct 16 different tasks. Those encompassed simple tasks like navigating and displaying information on the DS devices and more sophisticated ones, using the Update Program for updating content, changing modes and update plugins. Afterwards, the participants had to indicate the perceived simplicity for completing the tasks, using a 5-point Likert-scale ranging from $0 - 4$, indicating a task complexity from easy to very difficult.

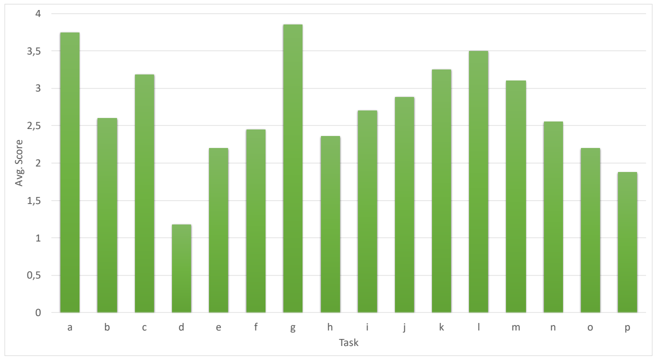


Figure 7: Average ratings of the participants for the 16 conducted tasks labeled as (a-p)

Results
As many people from the domain of computer science are comfortable with using touch-controlled devices, most of the tasks related to general navigation were perceived as easy to perform. The most problems were registered during the interaction with the DS running in the context of the Update Program. Figure 7 shows the average results for the conducted tasks labeled as (a) to (p). The currently implemented *action* for triggering update functionalities (i.e., right-click in contrast to left-click) was perceived as not intuitive and the absence of a functionality outline complicated matters. This is also reflected by the results of the tasks (d,e and f), which all required interacting with plugins using the Update Program. General remarks for improving the work were related to adding visual clues, tool-tips and feedback functionality that informs a user about successfully executing certain functionalities (e.g., successful update).

Conclusion and Future Work

Signage is transitioning from static analogue signs towards Digital Signage (DS) providing a variety of benefits in terms of maintenance, updateability and the support of dynamic content. In this context we presented the Modularly Operated Digital Signage (MODS) framework, facilitating the simple adaption of DS solutions towards diverse use-cases. This is done by organizing use-case specific solutions using plugins, which can then be loaded and orchestrated by the MODS framework. Finally, a pre-study was conducted to receive the first insights into the usability of the MODS. Overall, the usability of the framework was perceived as good, but additional visual clues and information about control functionalities are required to further improve the usability.

As an open and extensible framework, MODS provides a valuable basis for various future developments. Example directions that would especially benefit from the current design encompass context-aware functionalities. One can think of automatic updates regarding presence information, e.g., the digital sign can update itself when a person enters an office (marking the person as “available”) or if the person joins a telephone call (marking the person as “DND”). The way finding module becomes particularly useful if combined with a “meeting organizer” when it enables the routing of external guests. Combined with emerging sensors in the IoT, the usefulness of DS will further increase.

Acknowledgements

This work is based on the Software Campus project “ADRAS“, which is funded by the German Federal Ministry of Education and Research (BMBF) under grant no. “01IS17050‘ and also supported by the German Federal Ministry of Education and Research (BMBF) as well as by the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within CRISP.

REFERENCES

1. Colloqi Consulting. 2019. piSignage -Digital Signage for all. (2019). https://www.pisignage.com/, retrieved on 2019-05-17.

2. Goldmedia. 2019. Digital Signage becomes ubiquitous. (2019). https://www.goldmedia.com/en/news/info/article/digital-signage-becomes-ubiquitous/, retrieved on 2019-05-17.

3. John V Harrison and Anna Andrusiewicz. 2004. A virtual marketplace for advertising narrowcast over digital signage networks. *Electronic Commerce Research and Applications* 3, 2 (2004), 163–175.

4. Visix Inc. 2019a. Axis TV Signage Suite. (2019). https://www.visix.com/, retrieved on 2019-05-17.

5. XOGO Inc. 2019b. XOGO Decision Signage - Digital signage made easy. (2019). https://www.xogo.io/, retrieved on 2019-05-17.

6. Jon Knight and Jason Cooper. 2018. Raspberry Pi driven digital signage. (2018).