Bachelor's thesis

Information and Communications Technology

2019

Olavi Viitanen

INTEGRATING TWO DIGITAL SIGNAGE MANAGEMENT SYSTEMS

– Case: FirstView MediaCloud and Samsung MagicINFO

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communications Technology

2019 | 27 pages

Olavi Viitanen

INTEGRATING TWO DIGITAL SIGNAGE MANAGEMENT SYSTEMS

- Case: FirstView MediaCloud and Samsung MagicINFO

FirstView's MediaCloud is a digital signage management system that can be used to control displayed content through an internet browser. FirstView has commissioned the author of this thesis to plan the use of Samsung's MagicINFO digital signage management system to use its functionalities to remotely change display devices' input sources and power settings.

FirstView had an already installed version of the MagicINFO system prior to the thesis. The system was updated to the newest version. The server's configuration and certificates were verified for the deployment. Studying the use of the MagicINFO system was done by following the information found in documentation and by communication with a contact person at Samsung.

The thesis resulted in a prototype proving the integration is usable in production. Limitations were found on the MagicINFO service, which affects the usability. These limitations slow down the user experience and increase the required steps to set up new digital signage. Along with the requested functionalities, new use cases for the MagicINFO were found. With the integration's success, the potential for achieving the same functions with other digital signage manufacturers was mapped out.

KEYWORDS:

Digital signage, integration, management system, service interface

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikka

2019 | 27 sivua

Olavi Viitanen

KAHDEN DIGITAALISEN MAINOSNÄYTÖN OHJAUSJÄRJESTELMÄN YHDISTÄMINEN

- Case: FirstView MediaCloud ja Samsung MagicINFO

FirstViewin MediaCloud on digitaalisten mainosnäyttöjen ohjausjärjestelmä, joka mahdollistaa näyttöjen esittämän sisällön vaihtamisen internetselaimella. Yrityksen toivomuksena oli luoda suunnitelma Samsungin MagicINFO-palvelun käyttöönotolle. Ohjausjärjestelmien yhdistämisen tarkoituksena on tuoda MediaCloudiin mahdolliseksi muuttaa näyttölaitteen sisääntuloa ja virtatiloja. Nämä ominaisuudet parantaisivat asiakastuen toimintaa ja antaisivat tarkempaa tietoa mainosnäyttöjen tiloista.

Työtä varten päivitettiin FirstView'n tiloissa ennestään asennetut MagicINFO-palvelun järjestelmät. Palvelimen asetukset, sekä sertifikaatit tarkistettiin käyttöönottoa varten. MagicINFO-palvelun käytön selvittäminen tapahtui hyödyntäen sen dokumentaatiota, sekä varmistamalla asioita Samsungin yhteyshenkilöltä.

Opinnäytetyön tuloksena oli prototyyppi, joka todentaa palveluiden yhdistämisen käytettävyyden. MagicINFO-ohjelmiston palvelurajapinnan käytöstä löytyi vaatimuksia, jotka vaikuttavat palvelun käyttämiseen. Nämä vaatimukset hidastavat käyttökokemusta ja lisäävät työvaiheita uusien mainosnäyttöjen käyttöönottoon. Toivottujen ominaisuuksien lisäksi MagicINFOlle löydettiin myös muita käyttökohteita. Onnistuneen ohjausjärjestelmien yhdistämisen jälkeen selvitettiin mahdollisuudet toistaa toiminnot eri mainosnäyttöjen valmistajien palveluilla.

ASIASANAT:

Digitaalinen mainosnäyttö, yhdistäminen, ohjausjärjestelmä, palvelurajapinta

CONTENTS

FIGURES

LIST OF ABBREVIATIONS (OR) SYMBOLS

API	Application Programming Interface. Used for the communication between different programs or their parts.
CPU	Central Processing Unit. A part of a computer that handles most of the logic.
HDMI	High Definition Multimedia Interface. A cable type used to transfer audio and video between devices.
HTTP	Hypertext Transfer Protocol is used for the transfer of information through the internet.
HTTPS	Hypertext Transfer Protocol Secure. A secure version of HTTP where the information is encrypted.
LFD	Large Format Display. Refers to displays with screen sizes at or above generic televisions.
RAM	Random Access Memory. Storage space for limited data in small amounts requiring quick access.
RSS	Really Simple Syndication. It is used for publishing frequently updating information with a standardized format.
SoC	System on Chip refers to small integrated systems that function as computers.
SSL	Secure Sockets Layer. A protocol used for encrypting the information transferred through the internet.
SSSP	Samsung SMART Signage Platform is the embedded software used by Samsung to control their digital signage.
URL	Uniform Resource Locator. Forms a human readable address for a file location in a network.

1 INTRODUCTION

The current version of FirstView Ltd's digital signage management system is able to control the viewed content on the signage remotely through a web interface as long as the settings on the display are set correctly. Being able to change or view these settings would be useful for the customers themselves or for FirstView's support staff to better help guide the customer. One option for the functionality could be integrating Samsung's MagicINFO digital signage management system to the FirstView's MediaCloud and the main purpose of this thesis is to find out how feasible it is and plan the integration.

The theoretical part of the thesis will introduce the technologies used in the integration and the different parts of the MagicINFO solution. The practical part will focus on describing the integration process of setting up the MagicINFO Server and its use. The scope of the practical part is not to produce a completed feature, but to create a proof of concept that the solution will be usable in production once finished. At the end, the thesis will detail the plans for the continuation of the project and possibilities for other display manufacturers.

2 BASIC TECHNOLOGIES OF THE INTEGRATION

During the integration there are certain technologies that might require more explanation on what they are. This chapter focuses on giving a short general introduction to these technologies.

2.1 Digital Signage

Digital signage refers to electronic displays commonly used for displaying advertisements and information. There are many different types depending on the size and shape of the screen, the device controlling it, and the signage's interactivity.

The device controlling the screen can either be embedded in the digital signage itself or it can be an external computer connected through a network or small enough to be mounted behind the display. The device might be accessing a content management server to retrieve the display material, or it could be locally set on the device manually.

The sizes and shape can vary depending on the intended usage. The signage can be made to resemble a traditional foldable A frame sign, or it can be a huge installation covering an entire wall. Some signage benefit from being interactive by providing touch input for the user, for example, by changing from an advertisement to a map of a mall. Computer vision can be used to change the displayed media according to who is viewing the content. [1]

2.2 REST API

Representational State Transfer (REST) is a style of API design where the server is kept separate from the client. This gives the server the ability to respond to multiple clients and makes it easier to have the client and server handled by different organizations. Both client and server can then develop themselves independently. [2]

The second defining factor is that in REST, the server side is stateless in the communication. Every API request made should return the same answer every time they are made, and the answer should include everything in it to make it understandable. This

gives reliability to the service, but as a cost it can reduce network performance due to the fact that some data might be unnecessarily repeated. [2]

REST design also advises on uniform notation between calls to different resources. This helps users in understanding the system and makes it more scalable in future development. As a drawback, the network performance can be lower because the return values can contain unneccesary data. [2]

2.3 Spring Security and OAuth

Spring Security is a Java framework designed to handle authentication and access-control in a Java program. It is highly customizable for a project's demands and advertises integrations to other software. It has multiple options for authorization protocols, one of which is OAuth. [3]

OAuth is a protocol used specifically for authorization in HTTP communication. It is designed to be easily implemented on the client side [4]. The most notable part in it is the token which is given by the server in return for authentication. With the token the client can be given access to new features on the server. The token should be kept secure in an encrypted channel with HTTPS because others could use it malisciously.

Depending on the server's settings, the OAuth token can be a simple string with randomly generated characters, or it can be an encoded object containing information about the token. The tokens can have a time limit depending on how long they are active and on the user actions they can be invalidated ahead of time. A token should have no meaning on the client side, other than being an access key to a server. [5]

3 SAMSUNG MAGICINFO

Samsung's MagicINFO is their solution for digital signage displays and their management. It is divided in to three portions, which all have different options for different use cases. The MagicINFO Player includes the software that controls the displays themselves, while the MagicINFO Author is a tool to create the displayed content. The MagicINFO Server can be used to remotely control multiple MagicINFO Players. A graphical presentation of the relations can be viewed in Figure 1.

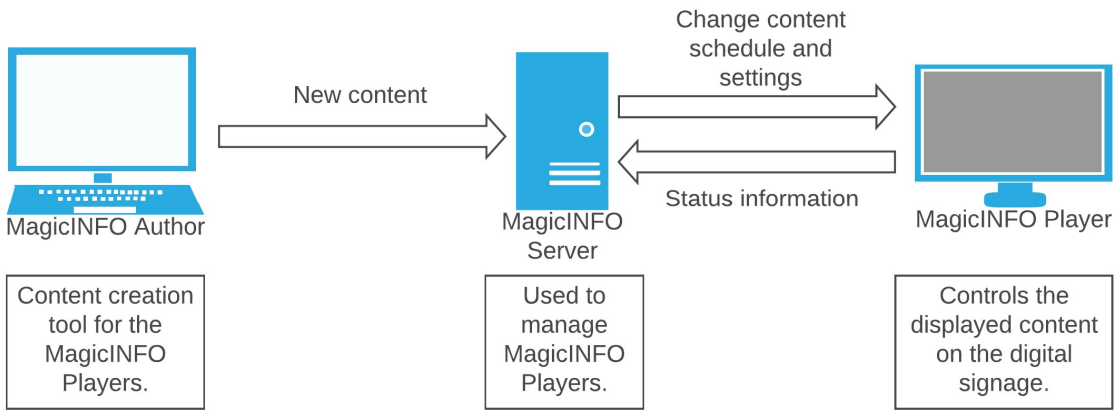

Figure 1. MagicINFO system's parts and their relations.

3.1 MagicINFO Player

The MagicINFO Player handles most of the functionalities of the Samsung's digital signage solution. It is responsible for storing the content and managing when and how to display the content on the screen. The MagicINFO Player has a few different versions based on how they connect to a display or what the features are. One of them is the A Player, which is an android application for Galaxy Tablets, but there is very little information about it, other than the fact that it enables the use of the tablet in the MagicINFO system.

The S Player is an SoC, meaning it is inside the display itself requiring no external devices to work [6]. The S Player uses the Samsung SMART Signage Platform (SSSP) which is their open-source solution for digital signage. [7] It is built on top the Tizen operating system that is based on Linux. The features of the S Player are dependent on the version of the SSSP inside the display.

The I Player is the more robust version capable of handling larger videowalls that need to be arranged in irregular patterns and content including complex animations and translucent layers. The I Player is an set-back box, which describes an external computer meant to be mounted behind a display. The I Player does not use the SSSP or Tizen but is built to run on a Windows operating system. [6]

The Lite Player is a lighter version of the S Player that can be used on less powerful systems, but offers a limited set of features. For example the only supported image file format is JPEG [8]. All Samsung products that support the S Player can also be changed to use the Lite Player [9]. This does not appear to have use cases when using the Player alone, but it gives an option to use a different licence when connected to a MagicINFO Server.

3.2 MagicINFO Server

The MagicINFO Server is used to remotely manage the MagicINFO Players that are connected to it. It gives access to the same settings the player itself has, but it also provides access to select settings in the displays themselves. The display settings can be changed in a quick menu, a more indepth menu or by using a virtual remote. The MagicINFO Server is able to edit multiple devices at the same time and you can set content and schedules to be shared between multiple destinations. This can be used to for example change a restaurant chain's menu signs in all locations.

While the MagicINFO Server itself and the use of MagicINFO Players is free, the connection between them requires active licences. The licences manage what features the MagicINFO Server and Players are able to use. If a display has an S Player installed, but the MagicINFO Server only has licences for the Lite Player, the display must be set to Lite Player mode to be connected to the server.

In addition to managing the MagicINFO Players, the server provides statistics from the displays and the server's actions. These include error logs from the devices, user actions on the server and a licence history. A summary of all the content, schedules and displays can be viewed in the front page of the service. If a MagicINFO Player supports it, extra information can be gathered about how effective the signage is with data on how long people view the displays and what age and gender group they belong to.

To use the MagicINFO Server, an account has to be created for it. Each account must be authorized by someone with administrator priviledges. Every user is assigned a role that limits what actions they are able to complete. Besides Administrator, the default roles include Device Manager, User Manager and Content Manager, each only able to access the functions for their managing area. One of the more limiting options is Content Uploader, which is only allowed to create content and playlists, but not edit them. Administrators are also allowed to create new user groups and decide creation, editing and deletion rights for every category one by one. [10] Users that are not Administrators can also be limited to access only certain devices.

The MagicINFO Server includes the Open API, which provides enough functionalities to rebuild most of the user interface of the management website. After an authentication to the API with the same credentials one would use for the MagicINFO Server, the user gets the same access to the system as they would get with the UI. For rebuilding the user interface, the API for example provides commands to list the available menu buttons and changing the order of listed items. [11] Missing features for the API include licence management and listing devices requesting authorization.

MagicINFO Server is also marketed as an another solution, called MagicINFO Remote Management. They are the exact same software and have no differences in their installation. The MagicINFO Remote Management only differs in what licence is given to an LFD. Since the functionalities of the MagicINFO Server are largely based on the licences are in use, both of these solutions can be used at the same time.

3.3 MagicINFO Author

MagicINFO Author is the content creation tool for Samsung’s LFDs. The Lite Players are not able to use the content created by it. The tool has three different versions, Premium Author and VideoWall Author are their own separate programs and the Web Author is a tool that comes with the MagicINFO Server. When creating content with the MagicINFO Author the target device’s software version has to be selected. This is to limit the functionalities to the ones that are supported by the target.

The Web Author provides a drag and drop interface with a selection of templates to guide on suggested placements of elements. The Web Author supports the integrated applications in the templates such as RSS feeds, local and external websites and viewing

a selected input from the display device. The ability to select content to be shown based on the device's tags is also possible through the Web Author. [6]

The Premium Author and Videowall Author become required when accurate control is required to change the viewed content. This setting is handled by using a timeline, where the user interface is similar to many video editors. Besides displaying and hiding content, its properties can be animated during set times. These properties include values for size, position and opacity. [12] The Premium Author is also needed for I Player when displaying content using VBScript or using touch controlled devices. The Videowall Author is for creating content for display set ups containing multiple screens. The Videowall Author also provides support for live camera and computer feed when using an I Player. [6]

4 INTEGRATION PROCESS

The main feature requested from the use of the MagicINFO solution, was the remote control of display devices from the FirstView MediaCloud. The content management and scheduling would be left to the MediaCloud. The features requested for implementation are toggling the display device's power, changing the source input and displaying status information of the display device itself. Any possible other features or ideas for the use of the MagicINFO were requested to be listed. These possibilities will not be made available to users in the first version, but their use was requested to be studied.

The starting point for the integration was not from a completely empty state. The Windows server to be used for hosting the MagicINFO Server was already running and an older version of the MagicINFO Server was installed on to it. The first task was to check in what state the environment was currently and then find out about the upgrading of the MagicINFO Server.

4.1 Updating the MagicINFO Server

The server in use is a Windows Server 2012 R2 standard, with an Intel Xeon E5-2676 v3 @ 2,40 GHz. The available disk space is 200 GB and RAM is 4,00 GB. The set up fulfills the recommendations for MagicINFO Server set by Samsung for around 200 clients. [13] The CPU is able to handle a larger load, but with enough growth the RAM would most likely be the first thing requiring an upgrade. The amount of disk space should not be a limit for the project, because the content handling will be handled by the FirstView's MediaCloud. Increasing the capacity to the next level of the recommendations will require restructuring of the server, since it suggests splitting up the web server and database to different machines.

The MagicINFO Server version already installed on the machine was v4.101.8 and the database in use was the PostgreSQL v9.3.8. Updating the MagicINFO Server to the v6.2.0 version had no difficulties, partially due to the fact that the database required no updates and no critical data was stored on it. MagicINFO provided a quick health check tool for the update and the update itself created logs of the process completion. Comparison of the logs from the update and from the initial installation ensured that nothing unexpected had happened.

After the upgrade the open ports needed to be checked to allow all of the required packets MagicINFO required. For the test environment everything was working, but the Amazon Web Services handling the public communication needed the ports 7002 and 990 to be opened for the basic features to work [13]. Rest of the ports were decided to be left closed, since their features were not going to be used in the initial phases of the project.

The original set up for MagicINFO Server did have SSL certificates already installed, but for the URL configuration in use at First Technology they needed to be changed with the certificates in use for MediaCloud. This required finding the licence storage location for the server and going through the Apache Tomcat configuration files. The configuration required changes due to it using Java Secure Socket Extension format for the certificates and the certificates supplied from First Technology were in Apache Portable Runtime format.

4.2 Licences

When initially accessing the MagicINFO Server it provides a limited set of functions. There is no mention of the content uploading and creating, device management or playlists. These functionalities will be hidden until a licence for a MagicINFO Player is added and activated in the server management settings. For the testing phase a set of limited time free licences were provided by Samsung.

The licence required for the remote operation of the devices is BW-RMS40SA, during testing the Lite Player's licence does not support the URL Launcher used for the displaying of the MediaCloud content. The S Player licence does support it as a form of content, but it disables it from simply switching to it as an input source on the display. The BW-RMS40SA licence is meant for the use case of remote management and it doesn't require the actual MagicINFO Player to be fully running on the device. This does in turn disable the MagicINFO Player from working, so any MagicINFO specific LFD content can not be used together with the MediaCloud.

4.3 Connecting a display to the server

The process of connecting the display to the server starts by changing the display's settings. In the menu at the System section the Play via setting must be set to URL launcher. This enables the use of MediaCloud. After that, in the Network section's Server Network Settings subsection is the configuration for the connection to the MagicINFO Server. The IP needs to be set to the one pointing to the MagicINFO Server and the Domain set to the used protocol. For this project it was HTTPS and changing this corrected the Port setting to the value of 7002. After this, the device should appear on the MagicINFO Server requesting authorization. To authorize the device, there has to be an activated unused licence on the server. After the authorization the connection process is ready, the server and device are able to communicate with each other.

These device settings would be good if they were already set before being shipped to a customer. This includes setting the URL launcher to the MediaCloud's address although it could be then remotely set after the MagicINFO connection has been made. The fact that the devices need to be authorized on the MagicINFO Server will most likely increase the amount of work done at the Firstview end. This depends on which solution for handling the devices and users is selected.

If the customers using the Firstview MediaCloud are given access to the MagicINFO Server, there needs to be a way to handle the fact that now there are two accounts for one user. The user rights need to be handled on both systems to prevent unauthorized access outside of their own organization. This method could give the users themselves the responsibility of managing their devices and licences. On the other hand, if a user was created for the MediaCloud to use as a whole and the API used only in a server to server communication the user rights could be handed completely on the MediaCloud. For example the user could only request information from a device only if they have access to it on the MediaCloud. This method would require the authorization of a device to be done by the Firstview employees, either before shipping or after the installation on the customer's side.

Discussion with the engineers at FirstView decided that using a shared user to access MagicINFO is the favored solution. With the licence management missing from the API, it would require the customers to learn how to use the MagicINFO Server's user interface.

The device settings can be set at the supplier, but authorizing the device will be done at FirstView after it has been physically installed at the customer's location.

4.4 API

The MagicINFO Server's Open API requires the user to have valid login credentials to the service, which are the same they would otherwise use to access the user interface. The authorization is done by calling /auth enpoint on the server's address. [11] It accepts POST method and the credentials are given in JSON format. The user ID is given with a key "username" and the password with "password". The call returns an authorization token to be used in all other calls and it expires at an unspecified time. A previous token does not expire if a new one is requested with the same credentials. This means that sharing the same user in the backend of MediaCloud would be functionally possible, as it would avoid race conditions when refreshing a token while someone else could be using the old one. Figure 2 shows the process of reauthenticating the shared user. The check for an old access token would be done on every API call to the MagicINFO Server.

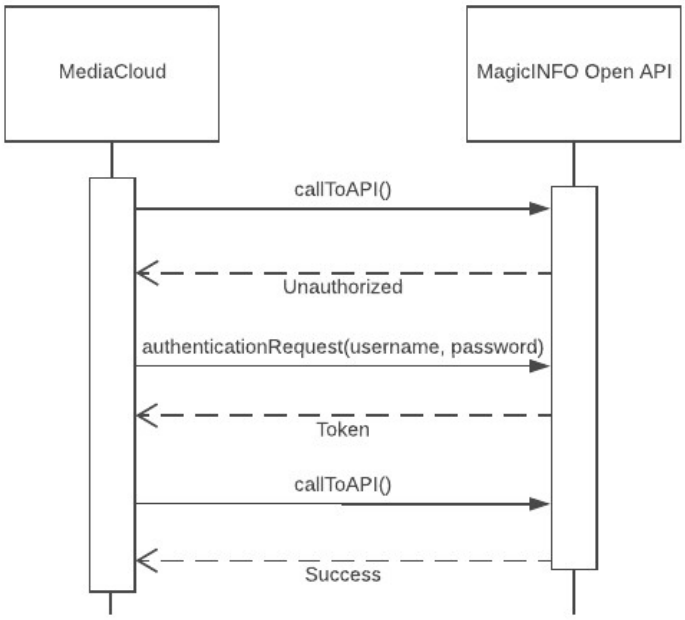

Figure 2. Refreshing authentication

Other calls to the API are done to a different endpoint which is /restapi/v1.0 [11]. The previously requested token is given in the header of the call as api_key. The

documentation for the API can be vague at times, for example only giving the data type of an option, but no information on what format it should be in. This has required a lot of testing to manually check what works and makes it harder to ensure it will work in every configuration. One point of confusion was the listing of valid input sources for a display changes in different calls, but a confirmation from Samsung ensures that the values used to select the input sources are the same for all display models [14]. This helps with being able to hard code the desired options to the MediaCloud.

Every call to the Open API has been encased in boilerplate code, to handle the possibility of a token being expired. The requests are inside try catch blocks to react to return statuses that are not the accepted value of 200. The failed request message is analyzed to decide if the reason was an old token or if something else went wrong on the MagicINFO Server. In the case of token being invalid, a new one is requested and the original request is sent again. The repeated request is also inside a try catch, but only to differentiate an error message to the user, it will not be tried again. If something unexpected has gone wrong on the MagicINFO Server, the exception is raised again to let it go as is to the user. Figure 3 contains the code for requesting a new token, while Figure 4 has the full boilerplate code.

```php
protected static $token;

static function requestToken() {
    $Client = new Client();
    $result = $Client->request(
        method: 'post', uri: self::MI_AUTH_URL,
        [
            'body' => json_encode(['username' =>  env( key: 'MAGICINFO_USERNAME'),
            'password' => env( key: 'MAGICINFO_PASSWORD')]),
            'headers' => ['content-type' => 'application/json']
        ]
    );

    $body = json_decode($result->getBody());

    if ($result->getStatusCode() == 200) {
        self::$token = $body->token;
    } else {
        throw new \Exception( message: "MagicInfo API Authentication Failed: " . $result->getStatusCode());
    }
}
```

Figure 3. PHP 7 code for requesting a new token from MagicINFO Server.

```php
public static function getRequest($endpoint) {
    $Client = new Client();
    try {
        // Try the request normally
        $result = $Client->request(
             method: 'get',
             uri: self::MI_API_URL.$endpoint,
            ["headers" => [ 'Accept' => 'application/json', 'api_key' => self::$token ]]
        );
        return $result;

    } catch (ClientException $clientException) {

        // This most likely means, the token is old
        $contentType = $clientException->getResponse()->getHeader( name: 'content-type');
        if ($contentType && $contentType[0] === "text/html;charset=utf-8") {

            try {
                // Request a new token and retry the request
                self::requestToken();
                $result = $Client->request(
                     method: 'get',
                     uri: self::MI_API_URL.$endpoint,
                    ["headers" => [ 'Accept' => 'application/json', 'api_key' => self::$token ]]
                );
                return $result;

            } catch (ClientException $exception) {

                // Token request most likely has failed
                $contentType = $exception->getResponse()->getHeader( name: 'content-type');
                if ($contentType && $contentType[0] === "text/html;charset=utf-8") {
                    throw new \Exception( message: "MagicInfo API Authentication Failed");
                }

                // Token was fine, but something else went wrong.
                throw new \Exception( message: 'Unauthorized command to MagicInfo');

            }
        }

        if ($clientException->getResponse()->getBody()->status === 'Fail') {
            throw new \Exception( message: 'Unauthorized command to MagicInfo');
        }
        throw $clientException;
    }
}
```

Figure 4. PHP 7 boilerplate code for HTTP GET requests.

Requesting information about a display device can be done in two ways. The first returns the saved information on the MagicINFO Server's database and the second gets up to date information from the display itself. The current display information is accessed by two API calls. The /restapi/v1.0/rms/deviceId/status/display endpoint returns a single use token called requestId for the second call. The /restapi/v1.0/deviceId/status/display/requestId endpoint gives the actual status information from the time that the first call was made. If these calls are made too quickly from each other, an error message is returned warning about device timeout exception. Through simple testing, the requests appear to work if there is five seconds between

them, with an increasing chance of failure if the time is below that. The required steps for requesting current display information are shown on Figure 5.This does not pose problems for the API design itself, but the calls to it must be limited on the front end side.

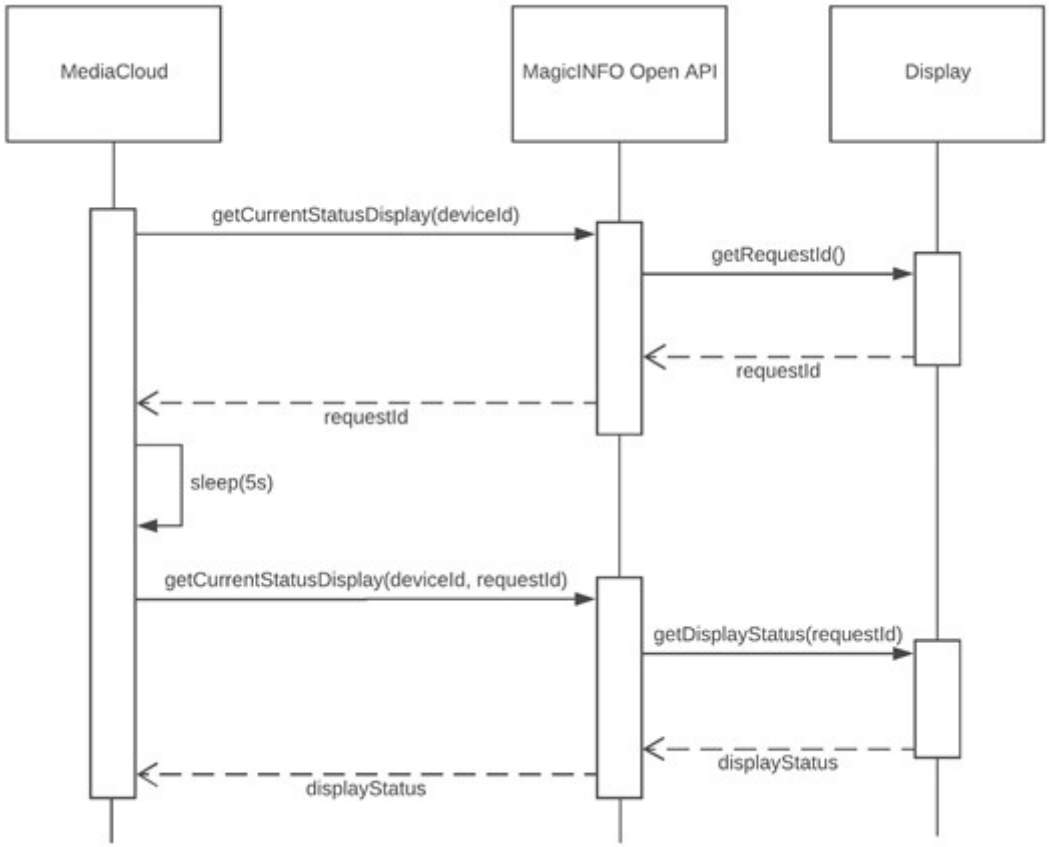

Figure 5. Sequence for current display status request

4.5 Security

Security requirements set for the integration include providing a secure connection for the communication and authorizing users for the API functionalities. While the MagicINFO Server's documentation focuses on providing examples in non-secure HTTP, it is not the only option. During the installation of the MagicINFO Server a selection is made whether to use HTTP or SSL. With the SSL option the server will only accept secured connections. Additional disabling for HTTP connections can also be made by not accepting anything from the port 7001 in the server's firewall and only open the port 7002 for HTTPS communication.

The authentication for the API calls on MagicINFO Server's end is handled by a Java framework called Spring Security. While the framework does support the use of OAuth protocol for the token creation, the configuration does not appear to be using it. The Spring Security configuration is custom made by Samsung. The configuration is similar to the OAuth protocol, where the authentication request returns a time-sensitive token that is then used in the header to access the rest of the API functionalities. The users are also authorized on the MediaCloud with OAuth 2.0, where their access rights are checked for the displays they are trying to connect to.

For smaller details on security, the credentials used to connect to the MagicINFO Server are not saved in the project's version management system and they have been selected to be long and unpronounceable. The user created for the API use has been given limited access rights to the MagicINFO Server to limit unwanted actions. Since the communication is done by server to server calls, the authorization token is never given to the user in the web browser.

5 FUTURE OF THE PROJECT

The original request for this thesis project was to bring the display device configuration functionality from the MagicINFO solution to the MediaCloud. With the basic structure of the API use completed, the practical portion's requirements have been met, but the project itself will be continuing by starting the design of the user interface for the display controls. Besides the practical part, the request was to map out other possibilities in the MagicINFO solution and find out if other display manufacturers offer similar functionality for their devices.

5.1 User Interface

While the creation and design of the user interface is not part of the scope, there are some design decisions in the API implementation that will affect on how it will be handled. The first design decision is the fact that updating the display status information is a lengthy process requiring at least five seconds of waiting. Updating the information should be kept to a minimum, where it is not done automatically after sending configuration changes or refreshing the page if there is previous data available. The MagicINFO Server's solution is to leave updating the information to the user by requiring a button press for each update. This will also give an option to show an error message guiding the user to try again after a moment if the display is unable to respond immediately.

The second design choice is brought to attention due to the wait time for the calls, because not all of the information about the display is gained from one call. The information is segmented to areas such as display, security and time. All of these have the waiting restraint in them. This could also be solved by implementing it using the same method as MagicINFO does , by only allowing the user to view one of the information types at a time.

With the decision to handle the communication through a single MagicINFO user, the creation of the links between MagicINFO and MediaCloud is left to the employees at FirstView. The licence management must be done on the MagicINFO Server itself, but MediaCloud will be used to add the MagicINFO device identification to the database. Authenticating a device on the MagicINFO Server could be done on either service, but

authenticating it on the MagicINFO Server when managing licences could be more streamlined.

5.2 Use cases

Before the project was started, there were no clear views on what the MagicINFO solution could achieve. The requested functionalities to change the input source and to toggle device power were fulfilled. One of the features found out during the working process is the ability to set the display to automatically shut down and start up at selected times. With the setting, one can also set the initial source for the display and the sound level. The days can be selected individually and a holiday period can be set where this feature is not used. Since this allows the display to shut down almost completely, only leaving the network on to listen for the start up command, electricity usage can be lowered, the display's life span is increased, and fewer calls will be made to MediaCloud reducing server load.

The security features in the displays were greater than expected and with the use of MagicINFO, the display can be almost completely locked from interacting with any other way than the API. This can be useful to prevent outsiders from changing channels or shutting down a display, while still giving easy access to settings and management for the authorized users. These features have to be made clear in their functions, since fully disabling the power button on a display can make it seem like the product is faulty, especially when the device is turned off.

While not strictly a way to use MagicINFO, the way the tagging system is handled is interesting. When an option is selected, content can be set to display on a device if both of them share the same tag. This can be useful in situations where a playlist is used in multiple different locations, but there are slight differences on what is needed to be displayed. For example, if a travel agency wants to have an information slide about the location in between advertisements, the slide can be selected based on a tag for the city. This functionality should be kept in mind, for possible future implementation in the MediaCloud.

5.3 Other manufacturers

One of the other manufacturers, First Techonology, uses is LG Electronics and they have their own remote management solution, SuperSign Control. The simpler version gives access to the basic settings of controlling power and volume, but it has a limitation of 100 displays. The advanced version called SuperSign Control+ has a wider range of settings and is able to view the display status in real time. [15] The solution is made with LG Electronics' own digital signage software webOS in mind.

The SuperSign Control+ Manager is used for the remote control and the SuperSign Control+ Agent handles the computation on the display device. Unlike MagicINFO, the Manager is also able to look for an Agent to connect to, but the Agent can also request connection by itself. [16] The documentation for SuperSign Control does not mention an API to control its features.

Philips, partnered with TeamViewer, offers a solution that gives full access to the display settings. Other functionalities include real-time performance information and viewing the display contents if the source is not set to HDMI. The solution requires the installation of TeamViewer Host on the display device, some models come with it already installed. The TeamViewer Host is assigned to a TeamViewer account by starting up the app and logging in on the display. [17] The solution does not directly offer an API, but TeamViewer does have REACH API for their integration partners. REACH API is designed to control the TeamViewer remote management solutions, but it is unclear if it can be used for the desired purpose.

While it might not be possible to integrate LG Electronics' and Philips' remote management options in the same way as MagicINFO, they could still prove useful. With approval from a customer, they could be used as customer support tools by using their own user interfaces. They could also be mentioned to customers requesting more remote control functionalities for their screens.

6 CONCLUSION

The thesis has resulted in an integration of Samsung's MagicINFO digital signage management system in to the FirstView's MediaCloud on the backend level. The integration brings remote management possibilities for suitable Samsung displays, with input source switching and power control being the main features requested.

The ability to remotely control screens is a helpful tool in customer support staff because they can change an input source quickly, eliminating the need to instruct a customer how to do it. The integration also provides information on the display's own status, making it possible to recognize situations where a display device is on, but the screen has been shut down. The status information also reduces the need to fully rely on a customer's description of the situation.

The limitations in the set up are the required wait times for the MagicINFO Server's API calls and the licence management. The wait times being multiple seconds per call can make the functionality feel slow and unreliable when users may be used to seeing results appearing in less than half a second. The licence management is an increase in work amount when setting up new devices to the system and they need to be maintained.

The set up used in the thesis can be useful in situations where remote management is desired, but there already is another solution in use for other areas related to displays and their management. If no other software is in use, a minimal benefit can be that the user interface can then be controlled and changed to fit the requirements.

REFERENCES

[1] Complete digital signage services [Internet]. FirstView. 2019 [cited 8 April 2019]. Available from: https://www.firstview.fi/en/solutions/

[2] Fielding R. Architectural Styles and the Design of Network-based Software Architectures [Ph.D]. University of California; 2000.

[3] Spring Projects [Internet]. Spring.io. [cited 8 April 2019]. Available from: https://spring.io/projects/spring-security#overview

[4] OAuth 2.0 — OAuth [Internet]. Oauth.net. [cited 8 April 2019]. Available from: https://oauth.net/2/

[5] Access Token Response - OAuth 2.0 Servers [Internet]. OAuth 2.0 Servers. [cited 8 April 2019]. Available from: https://www.oauth.com/oauth2-servers/access-tokens/access-token-response/

[6] Magicinfo | Digital Signage Software Solutions | Samsung Display Solutions [Internet]. Displaysolutions.samsung.com. [cited 8 April 2019]. Available from: https://displaysolutions.samsung.com/solutions/signage-solution/magicinfo/components

[7] SSSP | Software Solutions | Samsung Display Solutions [Internet]. Displaysolutions.samsung.com. [cited 8 April 2019]. Available from: https://displaysolutions.samsung.com/solutions/partner-solution/sssp

[8] Samsung. User Manual [Internet]. 1st ed. 2016 [cited 8 April 2019]. Available from: http://downloadcenter.samsung.com/content/UM/201604/20160404200359000/DCE_DCE-M_DCE-H_EDE_EME_NA_WebManual_Eng_US-00_20160329.0.pdf

[9] How to Choose Correct License for MagicInfo Server? | FAQ | Support | Samsung Display Solutions [Internet]. Displaysolutions.samsung.com. [cited 8 April 2019]. Available from: https://displaysolutions.samsung.com/support/faq/detail/how_to_choose_correct_license_for_magicinfo_server

[10] User role management - MagicINFO Server - samsung [Internet]. Displaysolutions.samsung.com. [cited 8 April 2019]. Available from: http://displaysolutions.samsung.com/docs/pages/viewpage.action?pageId=2065965

[11] Open API - MagicINFO Server - samsung [Internet]. Displaysolutions.samsung.com. [cited 8 April 2019]. Available from: http://displaysolutions.samsung.com/docs/display/MS1/Open+API

[12] Samsung. MagifInfo Author User Guide [Internet]. 1st ed. 2018 [cited 8 April 2019]. Available from: http://displaysolutions.samsung.com/docs/display/MA/Premium+Author?preview=/2067389/15864602/MagicInfo_Author_UM_Rev.1.0_Eng_180430.pdf

[13] Pettinen K. Samsung Remote Management. 2019.

[14] Hong S. query in relation to OpenAPI. 2019.

[15] SuperSign Control & Control+ | LG SuperSign Software | Software | Information Displ[ay | Business | LG Global [Internet]. Lg.com. [cited 8 April 2019]. Available from: https://www.lg.com/global/business/information-display/software/lg-supersign-software/supersign-control-control-plus

[16] LG Electronics. SuperSign Control+ Manual. 2019.

[17] TeamViewer Host and Philips Monitor integration [Internet]. community.teamviewer.com. [cited 8 April 2019]. Available from: https://community.teamviewer.com/t5/Knowledge-Base/TeamViewer-Host-and-Philips-Monitor-integration/ta-p/33858