SECURE. **NETWORKS.**

LANCOM Wireless ePaper Server

Integration Office 365

04/2019

LANCOM
Systems

Contents

Copyright

While the information in this manual has been compiled with great care, it may not be deemed an assurance of product characteristics. LANCOM Systems shall be liable only to the degree specified in the terms of sale and delivery.

This document contains statements relating to future products and their attributes. LANCOM Systems reserves the right to change these without notice. No liability for technical errors and/or omissions.

This product contains separate open-source software components which are subject to their own licenses, in particular the General Public License (GPL). The license information for the device firmware (LCOS) is available on the device's WEBconfig interface under "Extras > License information". If the respective license demands, the source files for the corresponding software components will be made available on a download server upon request.

Products from include software developed by the "OpenSSL Project" for use in the "OpenSSL Toolkit" (*www.openssl.org*).

Products from include cryptographic software written by Eric Young (*eay@cryptsoft.com*).

Products from LANCOM Systems include software developed by the NetBSD Foundation, Inc. and its contributors.

Products from LANCOM Systems contain the LZMA SDK developed by Igor Pavlov.

LANCOM Systems GmbH
Adenauerstr. 20/B2
52146 Wuerselen
Germany
www.lancom-systems.com

1 Objectives and requirements

ⓘ The instructions listed here are intended exclusively for experts! They also describe systems of a software version and are therefore for illustration purposes only.

1.1 Objectives

The objective of this document is to demonstrate how to successfully integrate the LANCOM Wireless ePaper solutions into a solution with Office 365 (Exchange Online). This is achieved with an IronPython script file, a configuration file, and an XSL template.

The code shown in the script file, configuration file and template are examples, which are intended to simplify the creation of your own files. If this code is to be used productively, then the information in the configuration file needs to be adjusted accordingly. Further templates can be created and used to optimize the adjustment to your intended usage. Furthermore the script file can be edited or rewritten if it does not meet your needs.

1.2 Requirements

The successful integration of the LANCOM Wireless ePaper Solutions into a Microsoft Office 365 environment requires certain conditions to be met with regards to the available software and the adjustments made to it. All three components run separately, but they do not necessarily need to be installed on different network resources. However, in the interests of system latency, the Wireless ePaper Server and the E-series access points should be connected over the same local network.

1. Script

   The following software is required on the computer running the script:

   - Microsoft Windows 7, Windows 8, Windows 8.1, Windows 10 (Tested: Windows 10 Enterprise 64 bit), Windows Server 2008 R2, Windows Server 2012
   - IronPython version 2.7.5 (tested)
   - Exchange Web Services Managed API version 2.2 which is available under

     https://www.microsoft.com/en-us/download/details.aspx?id=42951
   - .NET Framework 3.5 or higher

     ⓘ On some OSses .NET Framework 3.5 is already installed by default. Please check *this website* for an overview.

     The path to the installed DLL for the Microsoft Web Services Managed API needs to be specified in the environment variable `IRONPYTHONPATH`. Create the environment variable if one does not exist already.

     You can do this by rightclicking the computer icon on the desktop and select **Properties**. In the left click "Advanced system settings". In the dialog that opens select the tab **Advanced** and click **Environment Variables...**. In the dialog that open in the bottom part under System variables click **New**.

     As variable name enter `IRONPYTHONPATH`. Set the variable value to `C:\Program Files\Microsoft\Exchange\Web Services 2.2\`.

❗ Doublecheck the above variable value path in your filesystem to be sure you’re pointing tot he right location.

Then click **OK**.

2. Office 365

- An active Office 365 environment (Tested with Office 365 Enterprise E3)
- Windows Management Framework 4.0

❗ If you’re not running Windows 10 this is needed for being able to configure the Office 365 meeting rooms using Powershell.

3. LANCOM Wireless ePaper Server

The LANCOM Wireless ePaper Server needs to meet the following requirements:

- The template must be stored under \\<installation-directory>\data\templates\.
- Images referenced in the template must be stored under \\<installation-directory>\data\images\.

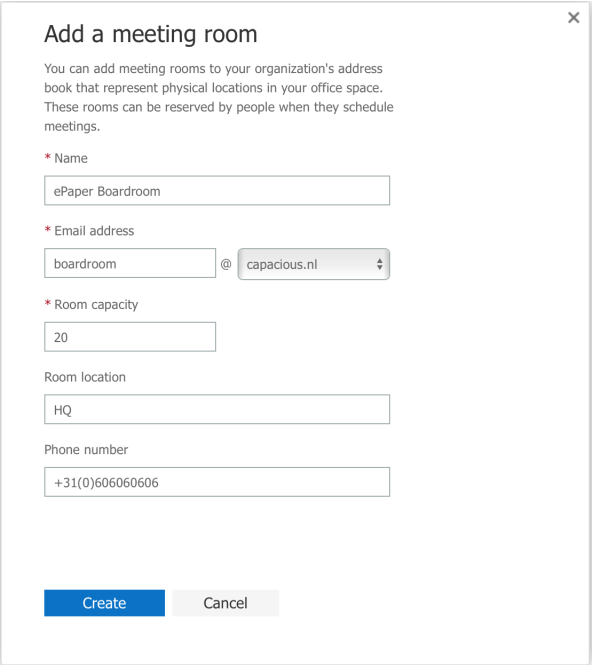
2 Adjustments in Office 365

Meeting rooms have their own mailboxes in Office 365 (Exchange Online), and the invitations to meetings arrive here. In order for the script to have access to these, a user with access to these mailboxes must be set up in Office 365.

Logon to *https://portal.office.com/* and navigate to the **Admin** console. In the left hand menu of the Office 365 admin center select **Meeting Rooms**. If not done yet create meeting rooms:

1. Click the "+" sign.
2. Enter a name for the meeting room.
3. Enter an email address for the meeting room.
4. Enter the room capacity.
5. Enter the room location.
6. Enter a phone number for the meeting room.
7. Click **Create**.

Note the name and the email address, you'll need to specify this in the `config.json` file. The other informations are not critical for the script.



Add a meeting room

You can add meeting rooms to your organization's address book that represent physical locations in your office space. These rooms can be reserved by people when they schedule meetings.

* Name: ePaper Boardroom

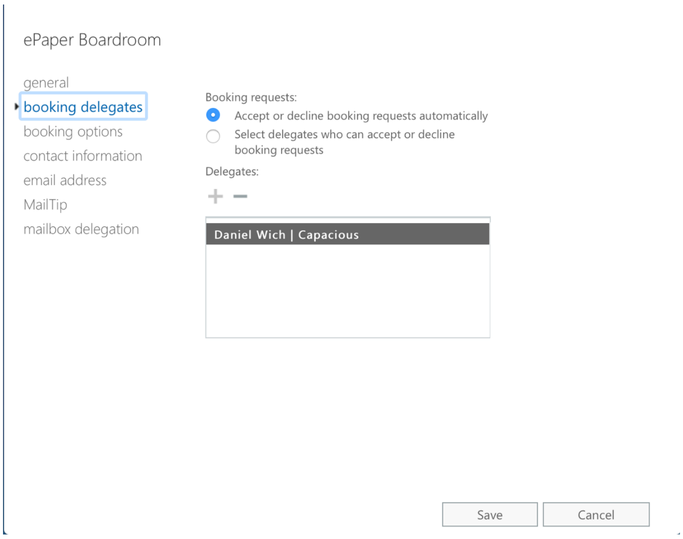
* Email address: boardroom @ capacious.nl

* Room capacity: 20

Room location: HQ

Phone number: +31(0)606060606

If already present select the properties of the meeting room. Then select the name of the meeting room and select **Edit** on the righthand side for more properties.



Booking requests:

● Accept or decline booking requests automatically

○ Select delegates who can accept or decline booking requests

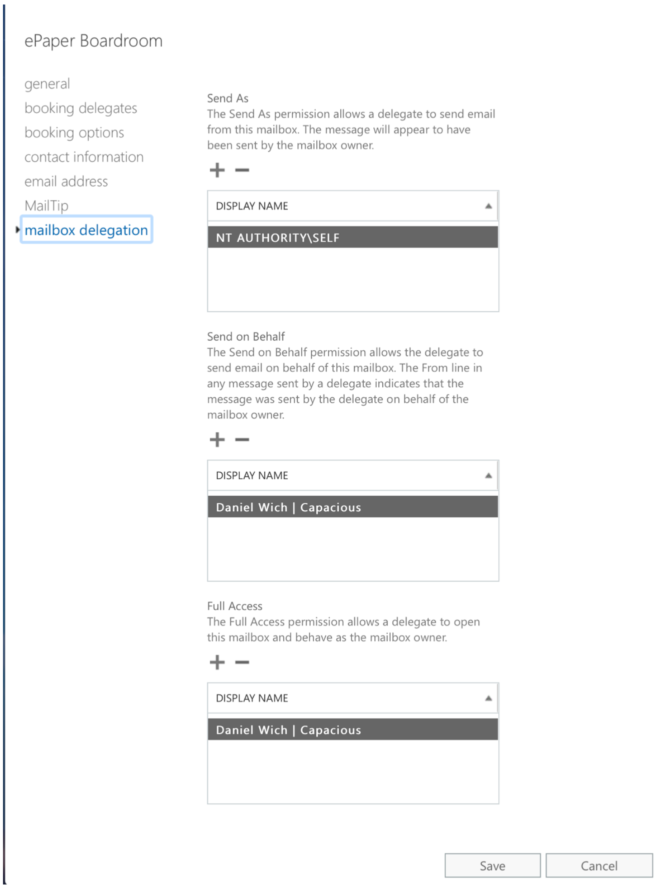
Delegates:

Daniel Wich | Capacious

Choose whether or not you want the mailbox to automatically accept and decline booking requests. Add a specific account as the Delegate of the mailbox. This account needs to be specified in the `config.json` file later on.

It might be handy to create a specific account in Office 365 for it but an existing administrative account also works.

Select **mailbox delegation** on the lefthand side:



Send As

The Send As permission allows a delegate to send email from this mailbox. The message will appear to have been sent by the mailbox owner.

DISPLAY NAME
NT AUTHORITY\SELF

Send on Behalf

The Send on Behalf permission allows the delegate to send email on behalf of this mailbox. The From line in any message sent by a delegate indicates that the message was sent by the delegate on behalf of the mailbox owner.

DISPLAY NAME
Daniel Wich | Capacious

Full Access

The Full Access permission allows a delegate to open this mailbox and behave as the mailbox owner.

DISPLAY NAME
Daniel Wich | Capacious

Add the specified account at the "Send on Behalf" and "Full Access" options. Then click **Save**.

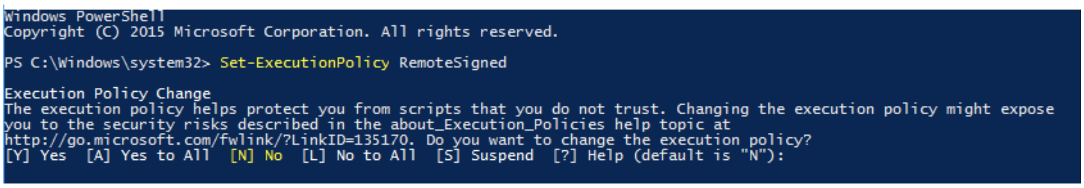
The Meeting Room has now almost been setup correctly, some Windows PowerShelling needs to be done to configure two last two critical settings, otherwise the meeting information on the display will not be correct:

- Set "Delete the subject" to **False**
- Set "Add the organizer's name to the subject" to **False**

For those who are not so experienced with Windows PowerShell the commands are as follows:

First you must start a Powershell under administrative permissions.

1. In Windows Search search for "powershell", rightclick it and select "Run as administrator" (**important!**).
2. At the prompt type `Set-ExecutionPolicy RemoteSigned`.
3. Type "`A`" (Yes to all) and press **Enter**.



```
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Set-ExecutionPolicy RemoteSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
http://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"):
```

Now you can execute the rest of the PowerShell commands that are needed.

To connect to Office 365 in the PowerShell console type the following commands:

- `$UserCredential = Get-Credential`

> ⓘ You'll be prompted for your credentials, in the example `daniel@capacious.nl` was used.

- `$Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri https://outlook.office365.com/powershell-liveid/ -Credential $UserCredential -Authentication Basic -AllowRedirection`
- `Import-PSSession $Session`

You should have a working PowerShell session connected to Office 365 now. If you want to lookup what version of Exchange is running in the cloud type `Get-OrganizationConfig | ft AdminDisplayVersion,IsUpgradingOrganization`.

In the `config.json` you'll need to specify an Exchange version EWS is going to try to use.

> ⓘ For this specification LANCOM recommends to use "Exchange2013_SP1".

To modify the parameters of the created meeting room use the following command:

```
Set-CalendarProcessing -identity 'ePaper Boardroom' -AddOrganizerToSubject
$false -DeleteSubject $false
```

> ⓘ Be sure that the identity is the same as the one that was used in the Office 365 admin center > Meeting Rooms configuration that was performed earlier!

Close the PowerShell session with `Remove-PSSession $Session`. Because of a limited number of PowerShell sessions that can be used at the same time be sure to close this session properly. Otherwise you'll have to wait for them to timeout before you can use a new session again.

3 Script and configuration file

The script and configuration file are a single logical entity, where the script file contains the program code and the configuration file specifies the parameters that vary from installation to installation.

3.1 Explanation of the script file

The script file `exchange_display_updater.py` contains the code that queries the relevant information from Office 365, processes it and, if necessary, sends an update to the LANCOM Wireless ePaper Server. The script file can be found on the *LANCOM Homepage* in the product area of the LANCOM Wireless ePaper Displays for download. No detailed explanation of the code is provided here, because all of the changes necessary for an integration are made to the configuration file.

3.2 Example of a configuration file

The configuration file `config.json` (any file name can be used) contains information about the LANCOM Wireless ePaper Server, the Office 365 Exchange Online Server, the template, and the Displays. It contains code in JSON notation and must be formatted as UTF-8 without BOM. Below is an example of a configuration file with comments to the individual entries.

```
{
    "wireless_display_server": {
        "address": "localhost",
        "port": 8001
    },
    "exchange_server":{
        "version": "Exchange2013_SP1",
        "default_credentials": false,
        "user": "daniel@capacious.nl",
        "password": "Delegate-01",
        "domain": "",
        "autodiscover_url_from_email": false,
        "autodiscover_email": "",
        "url": "https://outlook.office365.com/EWS/Exchange.asmx"
    },
    "conference_label": {
      "template": "lcsconference_landscape.xsl",
      "no_new_data_message": "No reservations"
    },
    "displays" : [
        {
            "exchange_room_mailbox": "boardroom@capacious.nl",
            "display_id": "D1011546",
            "display_name": "ePaper Boardroom"
        },
        {
            "exchange_room_mailbox": "conference-1@capacious.nl",
            "display_id": "D1010DEE",
            "display_name": "ePaper Conference Room 1"
        }
    ]
}
```

`wireless_display_server`

Specifies the LANCOM Wireless ePaper Server. All you need to enter here is the address in the LAN or localhost if the server runs on the same system.

`exchange_server`

Specifies the Microsoft Exchange Server.

`version`

Specified the version. This is important for the API. You can select from:

- `Exchange2007_SP1`
- `Exchange2010`
- `Exchange2010_SP1`
- `Exchange2010_SP2`
- `Exchange2013`
- `Exchange2013_SP1`

`default_credentials`

If this entry is set with the value `true`, the credentials of the current Windows user are taken for accessing the meeting-room mailboxes. If set to `false`, the values required are `user`, `password`, and `domain`.

The sample code presented here relies on the assumption that server communications take place either locally or on a trusted network. The network communication/access data are not especially protected.

`autodiscover_url_from_email`

If this entry is set to `true`, an email address from the domain can be used to find the Exchange Server. The email address is specified using `autodiscover_email`. If set to `false`, the Exchange service needs to be specified precisely by means of a URL.

`conference_label`

Specifies the basic properties of the meeting room signage.

`template`

Specifies the XSL template from the template folder.

`no_new_data_message`

Appears when there are no more reservations for the current day.

`displays`

Specifies the Wireless ePaper Displays being used. These are specified by means of a list of the individual Display objects. They have the following properties:

`exchange_room_mailbox`

The e-mail address of the meeting room mailbox

`display_id`

The display of ID (usually 8 digits in hexadecimal)

`display_name`

For the room name. This is fixed and is not affected by the room reservations.

4 Starting the update process

To start the update process, the script is started either from the command line, by batch file, or by service. The syntax for this is:

```
ipy.exe exchange_display_updater.py [-h] [-i <interval>] [-f] [-u (always|newdata|required)]
 [-d (debug|info|warning|error|critical)] <configfile>
```

What do these individual arguments mean?

-h

Displays help on the script and its arguments.

-i <interval>

Specifies the time interval between the individual checks in minutes.

-f

The first check will take place at the next full hour.

-u (always|newdata|required)

Specifies how the updates are carried out:

always

Every check sends an update to the Wireless ePaper Server in order to update the Displays. This should be used only for testing or with very long intervals to avoid unnecessary load on the Display batteries.

newdata

An update of the Displays via the Wireless ePaper Server is only triggered when the new data differs from that already on show.

required

Updates are similar to those triggered by `newdata`, except that the update takes place if the last update did not reach the SUCCESSFUL state. This setting is recommended if Displays are located at the edge of AP coverage.

-d (debug|info|warning|error|critical)

Specifies the debug level for internal logging. The default is `warning`: There is no output of the internal logging, except for the update activities.

-q

Disables the console output (quiet mode).

<configfile>

Specifies the configuration file (UTF-8 without BOM).

Example:

```
ipy.exe exchange_display_updater.py config_example.json -i 5 -u required
```

When the IronPython script starts, it first outputs the Exchange Server. From then on it continuously executes a loop that queries the Exchange Server for the meeting-room information. This is used to generate the content of the Displays, which is then sent to the Wireless ePaper Server depending on the selected update method.

Updates are interrupted with CTRL + C and a set of statistics is output to the console. The following is an example with an explanation for each item:

```
updates (new)                 :        2
updates (repeated)            :        3
updates finished successfully :        2
skipped                       :       11
```

updates (new)

Updates with new information sent to the Wireless ePaper Server.

updates (repeated)

Repeated updates sent to the Wireless ePaper Server after the previous update was not successful (only if the argument `-u required` is set).

updates finished successfully

Updates confirmed as successful (only if the argument `-u required` is set).

skipped

The update was skipped because of a lack of new information or the ongoing update still had the status WAITING.

5 Template for room signage

The template is stored on the Wireless ePaper Server as an XSL file. It specifies the format in which the Wireless ePaper Server renders the updates sent by the script library. The resulting image is then delivered to the corresponding Wireless ePaper Displays.

5.1 Example of a template

This XSL template is based on a potential meeting-room signage concept xml:lang="en"for the LANCOM Systems GmbH. Line numbering is not a part of the code and is for purposes of clarity only.

```
<?xml version="1.0" encoding="UTF-8"?>
<!—The template is provided for 7.4" Displays mounted in the landscape orientation.-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="Record">

 <!-- Rendering information for 7.4” Displays -->
 <image height="480" width="800" rotation="90" font-family="Verdana">

  <!-- Room -->
  <field height="108" width="780" x="10" y="20">
   <text align="center" font-size="40" font-weight="bold">
    <utils method="toUpperCase">
     <xsl:value-of select="room/@roomName"/>
    </utils>
   </text>

    <!-- Date -->
   <text align="center" font-size="35" font-weight="bold" padding-top="10">
    <xsl:value-of select="room/field[@key='date']/@value"/>
   </text>
  </field>

  <line thickness="2" x-from="0" x-to="800" y-from="130" y-to="130"/>

  <!-- Time1 -->
  <field height="50" width="780" x="20" y="150">
   <text align="left" font-weight="bold" font-size="40">
    <xsl:value-of select="room/field[@key='time1']/@value"/>
   </text>
  </field>

  <!-- Purpose1 -->
  <field height="50" width="780" x="20" y="200">
   <text align="left" font-size="40" condense="1, 0.8, 0.6, 0.5">
    <xsl:value-of select="room/field[@key='purpose1']/@value"/>
   </text>
  </field>

  <!-- Chair1 -->
  <field height="40" width="770" x="20" y="250">
   <text align="left" font-weight="bold" font-size="30">
    <xsl:value-of select="room/field[@key='chair1']/@value"/>
   </text>
  </field>
```

```

   <!-- Time2 -->
  <field height="35" width="780" x="20" y="320">
   <text align="left" font-size="28">
    <xsl:value-of select="room/field[@key='time2']/@value"/>
   </text>
  </field>

  <!-- Purpose2 -->
  <field height="35" width="780" x="20" y="355">
   <text align="left" font-size="28" condense="1, 0.8, 0.6, 0.5">
    <xsl:value-of select="room/field[@key='purpose2']/@value"/>
   </text>
  </field>

   <!-- Chair2 -->
  <field height="30" width="770" x="20" y="390">
   <text align="left" font-size="20">
    <xsl:value-of select="room/field[@key='chair2']/@value"/>
   </text>
  </field>

  <!-- LANCOM Logo -->
  <field align="right" height="60" width="780" x="10" y="410">
   <img src="../images/LANCOM_Logo_200.png"></img>
  </field>
 </image>
</xsl:template>
</xsl:stylesheet>
```

5.2 Explanations of individual sections of code

This chapter explains the various sections of code in the example XSL template.

Display information

```
  <!-- Rendering information for 7.4" Displays -->
  <image height="480" width="800" rotation="90" font-family="Verdana">
```

This section specifies which Wireless ePaper Display the following code applies to. In this case, it is a 7.4" Display with a resolution of 800 x 480 pixels.

`height` specifies the height of the image in pixels.

`width` specifies the width of the image in pixels.

`rotation` specifies the rotation of the image in degrees. 0 corresponds to vertically mounted Displays and 90 to horizontally mounted Displays.

`font-family` specifies the font to be used.

Definition of text fields

```
  <!-- Room -->
  <field height="108" width="780" x="10" y="20">
   <text align="center" font-size="40" font-weight="bold">
    <utils method="toUpperCase">
     <xsl:value-of select="room/@roomName"/>
    </utils>
   </text>

```

```
 <!-- Date -->
 <text align="center" font-size="35" font-weight="bold" padding-top="10">
  <xsl:value-of select="room/field[@key='date']/@value"/>
 </text>
</field>
```

This area specifies the position of the room name and date.

First, a field is defined that specifies the height, width and position on the Display:

```
<field height="108" width="780" x="10" y="20">.
```

`height` specifies the height of the field in pixels.

`width` specifies the width of the field in pixels.

`x` specifies the distance from the left edge of the Display in pixels.

`y` specifies the distance from the top edge of the Display in pixels.

The layout of the various texts is defined in this field:

```
<text align="center" font-size="40" font-weight="bold">
```

`align` specifies how the text is justified (e.g., center, left, right).

`font-size` specifies the font size.

`font-weight` specifies the font style (e.g., bold, italic).

```
<utils method="toUpperCase">
```

Invoking this method forces the text to appear in the uppercase.

```
<xsl:value-of select="room/@roomName"/>
```

Here, the XML information provided by the script library is accessed and the text is inserted at the appropriate position. So far, the definitions have only set out the layout.

The next step uses the same principle to specify the text that shows the transmitted date. Additional fields contain the layout for the rest of the information that relates to the individual meetings.

Integration of graphics

```
<!-- LANCOM Logo -->
<field align="right" height="60" width="780" x="10" y="410">
 <img src="../images/LANCOM_Logo_200.png"></img>
</field>
```

In addition to text fields, images can be integrated too. Similar to the above, a field is specified that sets the position of the image.

```
<img src="../images/LANCOM_Logo_200.png" ></img>
```

The storage location and the name of the image file is specified here.

5.3 Interaction between template, XML information and Wireless ePaper Displays

This chapter uses an example to describe the interaction between the template, XML information and the actual representation on the Wireless ePaper Display.

Step 1:

When an update is triggered by the Notes agent, the Wireless ePaper Server receives XML information with the following content: (Line numbering is not a part of the code and is for purposes of clarity only.)

```
001 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
002 <TaskOrder title="Refresh D1001BF6 for Aachen B2.100">
003   <TemplateTask  xml:lang="en"labelId="D1001BF6" externalId="4711"
        template="lcsconference_landscape.xsl">
004     <room roomName="Aachen B2.100">
005       <field key="date" value="16.09.2014"/>
006       <field key="time1" value="10:00 - 11:30"/>
007       <field key="purpose1" value="Marketing Project Coordination"/>
008       <field key="chair1" value="Alec Coad"/>
009       <field key="time2" value="11:30 - 13.00"/>
010       <field key="purpose2" value="Team Meeting Controlling"/>
011       <field key="chair2" value="Emily Kirby"/>
012     </room>
013   </TemplateTask>
014 </TaskOrder>
```

Step 2:

The Wireless ePaper Server processes the received information line by line. This sets the relevant Display and the template required. Based on the template, the Server now creates the data set used to render the image. The layout is set according to the template and then the information from the XML is inserted.

Two content elements are described here for illustrative purposes: The chairperson of the next meeting and the company logo.

The chairperson for the subsequent meeting is formatted according to the template as follows:

```
061   <!-- Chair2 -->
062   <field height="30" width="770" x="20" y="390">
063    <text align="left" font-size="20">
064     <xsl:value-of select="room/field[@key='chair2']/@value"/>
065    </text>
066   </field>
```

The information necessary for the representation can be seen in the XML below:

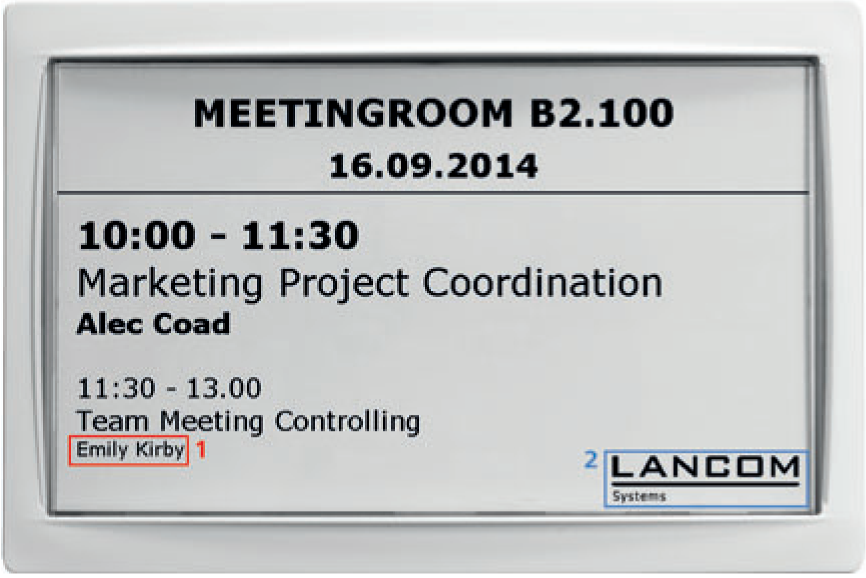
```
011    <field key="chair2" value="Emily Kirby"/>
```

Access to the image specified in the template does not require access to the XML information. The path is evaluated instead.

```
068    <!-- LANCOM Logo -->
069    <field align="right" height="60" width="780" x="10" y="410">
070     <img src="../images/LANCOM_Logo_200.png"></img>
071    </field>
```

Step 3:

The image is rendered and sent to the Wireless ePaper Display. The Display appears as shown below. The chairperson of the subsequent meeting **(1)** and the image **(2)** are highlighted in color.



MEETINGROOM B2.100

16.09.2014

10:00 - 11:30

Marketing Project Coordination

Alec Coad

11:30 - 13.00

Team Meeting Controlling

Emily Kirby