Design Guide SESAME-S Tablet

Author: Ulrich Lehner
2011-12-20

Creating
Communication
Technologies

1 INTRODUCTION 3
1.1 UNITS 3
2 FRAGMENTS 3
2.1 COMMON FEATURES 3
2.2 REAL TIME ENERGY CONSUMPTION 3
2.2.1 Input parameters 4
2.2.2 Display parameters 4
2.2.3 Output 4
2.2.4 Extended features 4
2.2.5 Mockup 5
2.3 REAL TIME OVERALL CONSUMPTION 5
2.3.1 Input parameters 5
2.3.2 Display parameters 5
2.3.3 Output 5
2.3.4 Mockup 5
2.4 REAL TIME LINE CHART 5
2.4.1 Input parameters 5
2.4.2 Display parameters 5
2.4.3 Output 6
2.4.4 Extended Features 6
2.4.5 Interaction 6
2.4.6 Mockup 7
2.5 OVERALL REAL TIME LINE CHART 8
2.5.1 Input parameters 8
2.5.2 Display parameters 8
2.5.3 Output 8
2.5.4 Extended Features 8
2.5.5 Interaction 9
2.5.6 Mockup 9
2.6 SUCCESS INDICATOR 9
2.6.1 Input parameters 9
2.6.2 Display parameters 9
2.6.3 Mockup 9
2.7 POWER MANAGEMENT SERVICE 10
2.8 AMBIENT SIGNAL 10
2.8.1 Display parameters 10
3 LAYOUTS 11

1 Introduction

This document is a style guide for the energy feedback monitor as part of the SESAME-S project. The overall goal for this app is to provide an energy feedback monitor and the integration for the Power Management Service. Since the test trials will compare different configurations of the app (e.g. detailed vs. abstract, small screen vs. large screen, etc.) the final presentations should be a composition of different widgets (further stated as *fragments*) defined by a *layout* (e.g. more detailed view, displayed on a Smartphone, etc.).

1.1 Units

The provided unit for energy consumption from the cloud service provided by E-Smart is (as known so far) Watt. Since for an average person this value may be meaningless an alternative conversion into a currency should be given (Euro). This might not be that trivial, since converting depends e.g. on the provider or the time (day vs. night), but this information should be available in some form (maybe E-Smart?). However the app should be prepared to show following units:

1. **kW** as the currently needed energy
2. **kWh** as the consumption so far
3. **Euro** as money spent so far or (for real time values) and as the electric price for the current consumption (again this must be clarified)

2 Fragments

The app will mainly be a combination of different fragments displaying data about energy consumption or allowing users to interact with the device or services. Most of these fragments can be visualized with the library *AChartEngine*[1] for the Android platform. Some of these fragments may be displayed with more options (referred as *extended features*, e.g. displaying a historical values).

2.1 Common features

All fragments should be designed in a way that they can consist in an own frame (e.g. to style it with a border, background color etc.). Analog to a UI window a fragment should have a title displayed at the top.

2.2 Real time energy consumption

This fragment should visualize real time consumption in form of a dial chart (like the metaphor of an analog tachometer, as shown in Figure 1 Tachometer). It is intended to give a real time feedback of the current energy consumption without distracting from other values. In the case of BBS Kirchdorf real time means ’15 minute intervals’, so changes from one value to another will be very discrete (this may look like the hand will jump from value to another). However to avoid this situation, a value change should be animated (since this user control will be customized anyway, responsibility of this behavior will be at FTW).

[1] http://www.achartengine.org/

Figure 1 Tachometer

2.2.1 Input parameters

The latest measured LoadProfile (*DataValue*) from a cloud service for a specific room (e.g. *EDV1*) – since the parameters require a time range this must be tricked somehow.

2.2.2 Display parameters

1. Color for a handle (simplified as constants like *RED* or *BLUE*)
2. To determine the range following attempts are possible (since this fragment should be displayed for each room the ranges should be the same for all)
    a. *Static* (the value of 10% is just an assumption, there may be better values)
        i. Lower bound: min value of overall consumption of all rooms – 10% rounded to a 'pretty' value
        ii. Upper bound: same as above from max side + 10%
    b. *Dynamic*
       Same rules as above but done programmatically
3. Unit of displayed data
4. Label for the room name

2.2.3 Output

The latest received *DataValue* will be displayed as value (handle position) with a *DialChart*. For further visual enhancement this chart will be extended to display the chart in a more elegant way, but the functionality remains the same.

2.2.4 Extended features

For more detailed information a second value should be displayed (in a less dominant col-or/style then the main handle) to compare the consumption. Following values are possible and should be adjustable:

1. Overall average consumption of a room from the beginning of a given date (in case of BBS Kirchdorf the begin of the trial)
2. Consumption of the previous (labor) day at the same time (e.g. now=2011-12-20 17:00 then consumption of 2011-12-19 17:00)
3. Consumption of the same day at the previous week at the same time (e.g. now=2011-12-20 17:00 then consumption of 2011-12-13 17:00)

2.2.5 Mockup

See Figure 1 Tachometer.

2.3 Real time overall consumption

To provide information of how much energy was consumed from a given date (in this case the start of a test trial), a 'power meter'-like visualization should be implemented. This simply gives a nice overview and should later be styled to look more compelling (comparing Figure 2 Digits from a power meter this gives a realistic look of the digits). For the first iteration of implementation this should be done in simplified way, e.g. in a *TextBox* (which will later be styled, derived etc.). A change of a digit should be animated (responsibility of FTW).

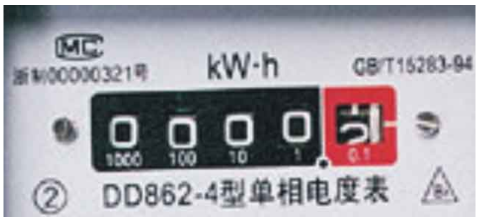

Figure 2 Digits from a power meter

2.3.1 Input parameters

This fragment should receive its data from the LoadProfile (maybe DailyConsumption will be the better choice) from a given date (should be flexible, but probably the start date of a trial) to the current time and sum up the *DataValues*.

2.3.2 Display parameters

1. Unit of displayed data.

2.3.3 Output

Current consumption

2.3.4 Mockup

See Figure 2 Digits from a power meter.

2.4 Real time line chart

This fragment should visualize the real time consumption on a daily basis displayed as a line chart for a specific room.

2.4.1 Input parameters

Data should be easily accessed via the *LoadProfile* function for a specific room for the current day.

2.4.2 Display parameters

1. Time range (basically it makes sense to set these bounds a little bit lower/higher then a regular school day, e.g. from 7am to 8pm)
2. Room (Measure Point)
3. Unit of displayed data

2.4.3 Output

This fragment shows the real time consumption (see Figure 3 Simple line chart) and optionally previous consumption (see Figure 4 Comparing line chart and Figure 5 Combined line chart) for a better understanding.

2.4.4 Extended Features

It is also interesting to compare the current situation with past consumptions. This can be implemented in the following ways:

1. On a per-day basis
    a. *Comparing one day:* consumption of previous (labor) day. This can visualized by drawing a second graph behind the actual in a less dominant way (e.g. make it less opaque). See Figure 4 Comparing line chart.
    b. *Comparing more days:* Consumption of previous (labor) days. Can be achieved the same way as above (layering graphs and decrease opaque – this of course is limited soon). See Figure 5 Combined line chart.
2. On a per-week basis
    a. *Comparing previous week:* consumption of previous week, displayed as above
    b. *Comparing more weeks:* consumption of previous weeks, displayed as above

2.4.5 Interaction

In case of this fragment an interaction from the user is required to display the chart of a specific room. This should be accessible by a *TabView* or *Buttons* to select a room. This interaction may motivate the user to actively engage with the app (see Figure 3 Simple line chart and Figure 4 Comparing line chart).

Also to give more possibilities to compare different values (in case of the extended feature set 1a and 2a) a *DropDown* list should allow the user to set the comparing value (in case of 1a a list of previous daya respectively a list of previous weeks in 2a) – see Figure 4 Comparing line chart. Cases 1b and 2b are intentionally combined solutions and shouldn't offer the user to choose.

2.4.6 Mockup

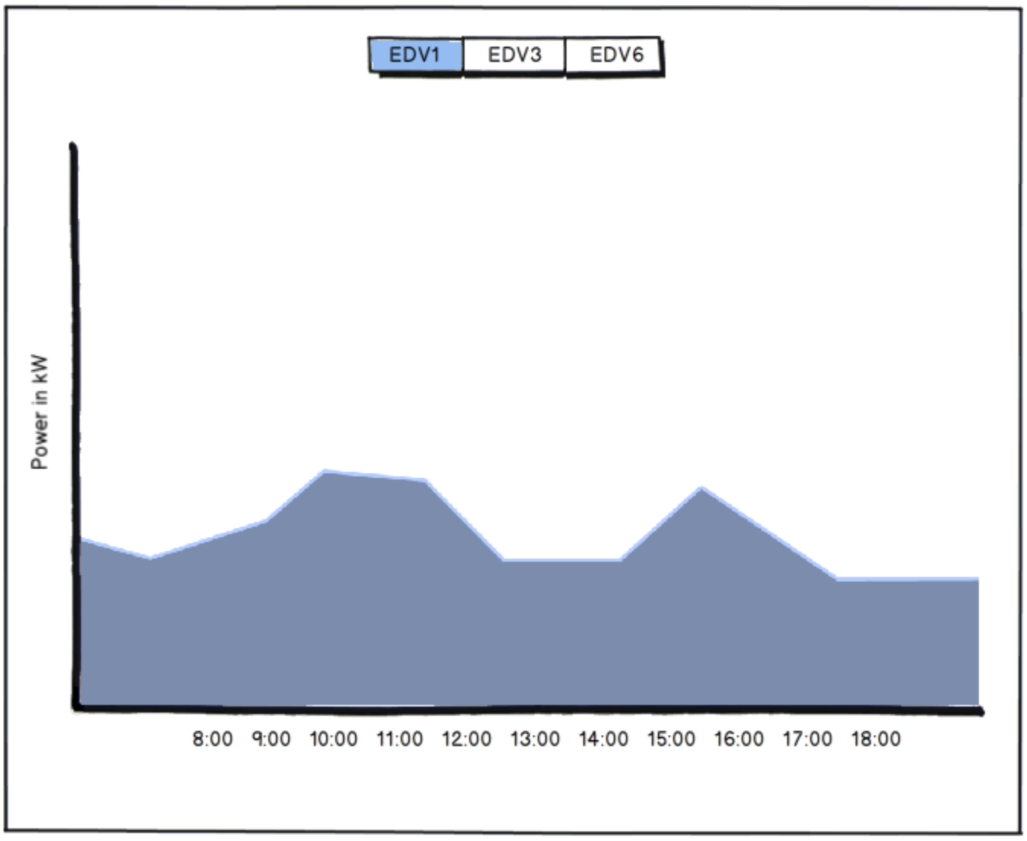

Figure 3 Simple line chart

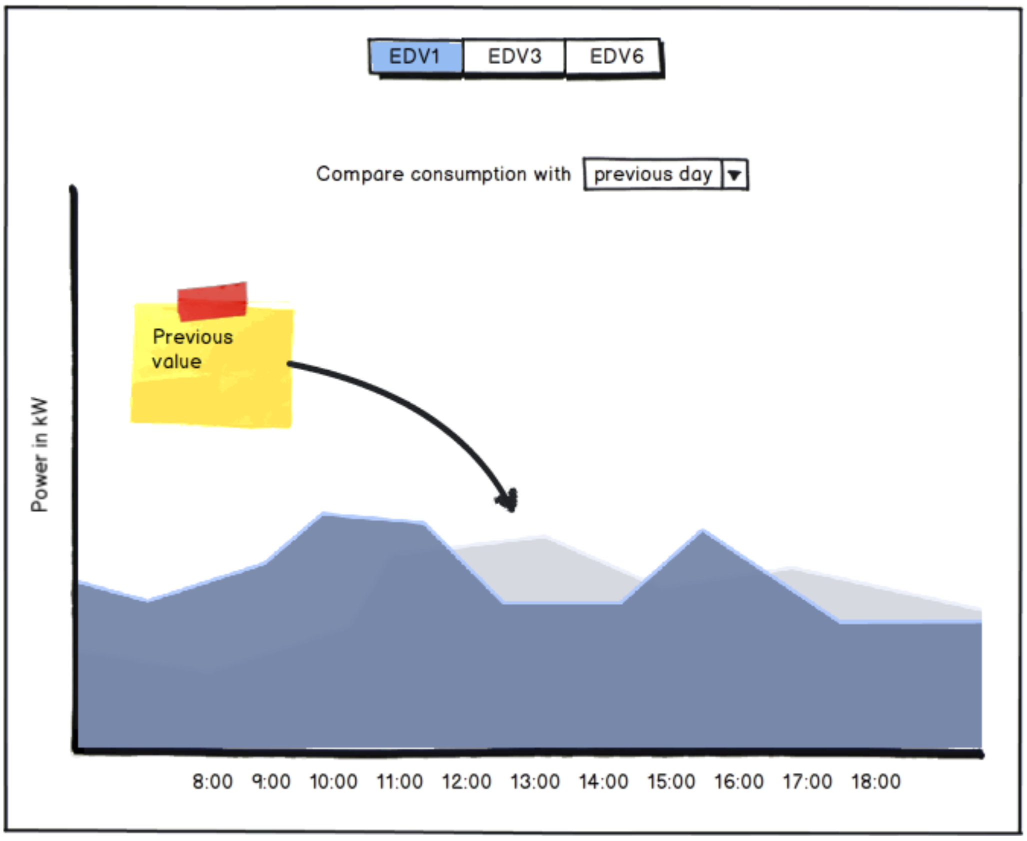

Figure 4 Comparing line chart

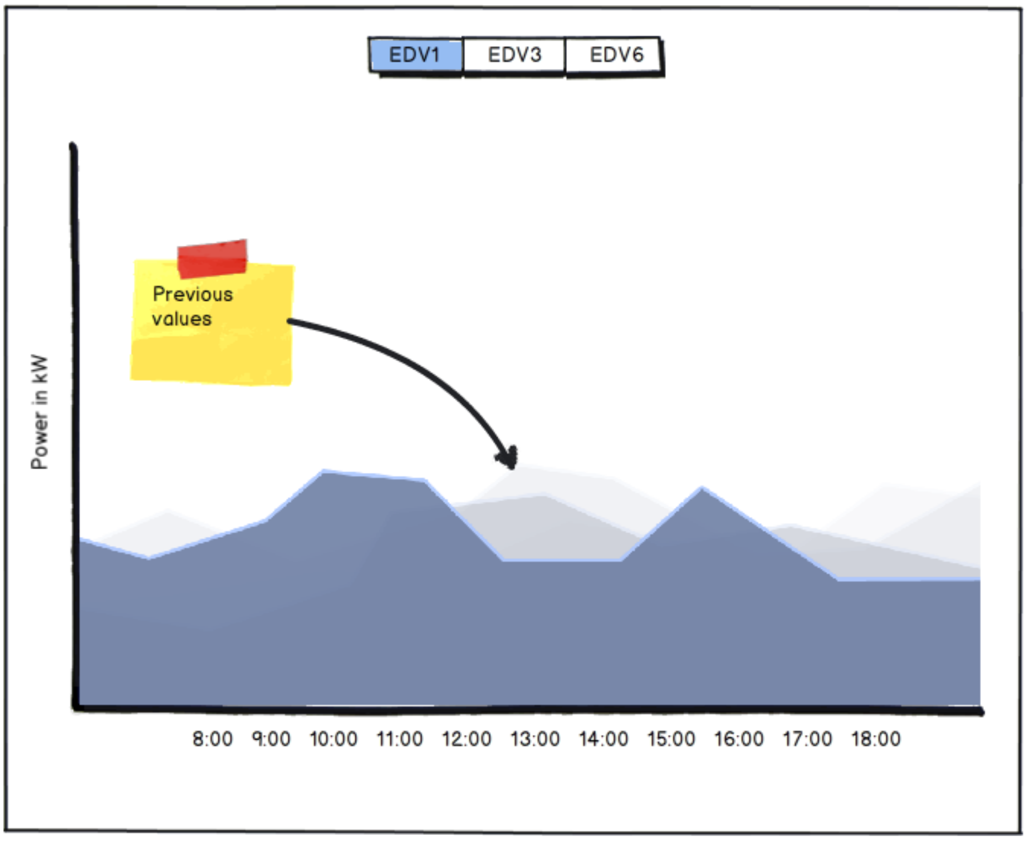

Figure 5 Combined line chart

2.5 Overall real time line chart

This is basically the same visualization as 2.4 Real time line chart with the difference that there is no distinction between the rooms (Measure Points). Instead one line graph should be displayed with the average consumptions of all rooms. With this simplification, the requirements for this fragment are a the same as in 2.4 Real time line chart and therefore it may be efficient to combine these features (since the comparison methods a actually the same).

2.5.1 Input parameters

Average *LoadProfile* of all *MeasurePoints* for the current day.

2.5.2 Display parameters

See above (except MeasurePoint)

2.5.3 Output

See above.

2.5.4 Extended Features

Since the consumption is composed of each room, an option to see the itemized consumption would be a nice feature to attract attention. See Figure 6 Compound line graph.

2.5.5 Interaction

By pressing onto the line a popup containing the information of the specific x value should be displayed (e.g. as bar chart or pie chart). See Figure 6 Compound line graph.

2.5.6 Mockup

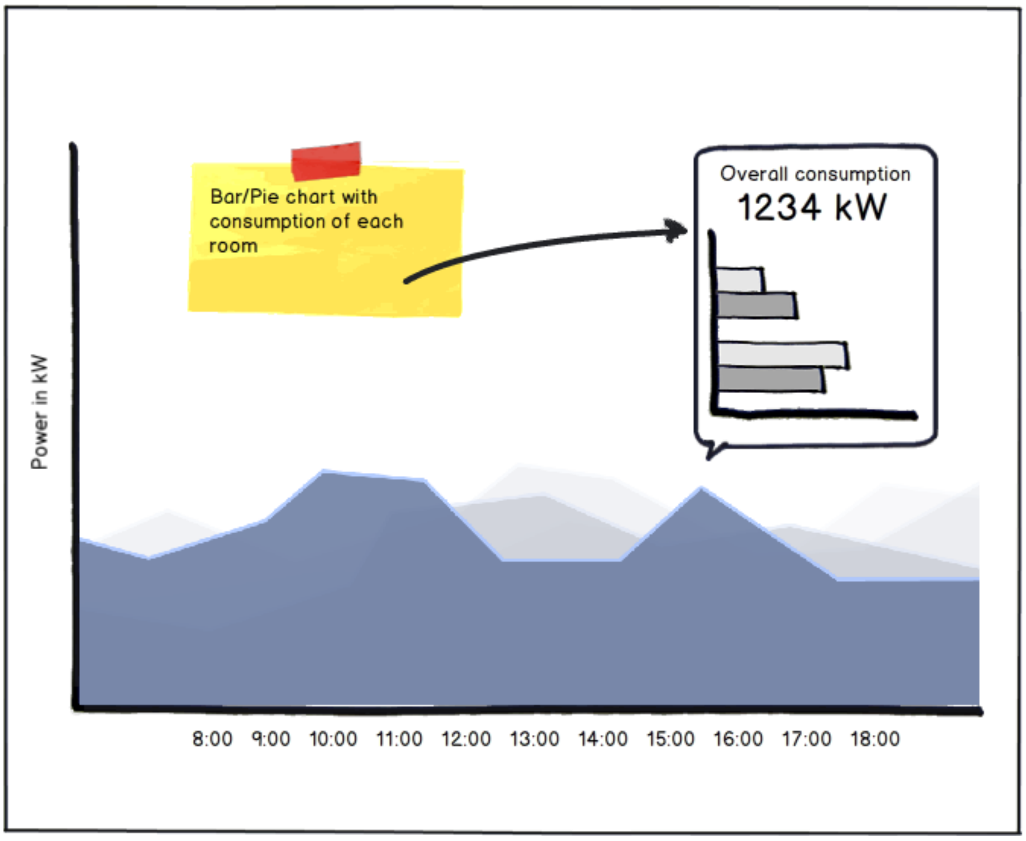

Figure 6 Compound line graph

2.6 Success indicator

This fragment is supposed to be combined with a line chart, showing how the current behavior is performing (e.g. current consumption is beneath the average consumption will result in a positive indicator vice-versa). It should be able to display three states: positive, neutral and negative (see Figure 7 Success indicator).

2.6.1 Input parameters

The input should correspond to 2.4 Real time line chart respectively 0 It is also interesting to compare the current situation with past consumptions. This can be implemented in the following ways:

3. On a per-day basis
    a. *Comparing one day:* consumption of previous (labor) day. This can visualized by drawing a second graph behind the actual in a less dominant way (e.g. make it less opaque). See Figure 4 Comparing line chart.
    b. *Comparing more days:* Consumption of previous (labor) days. Can be achieved the same way as above (layering graphs and decrease opaque – this of course is limited soon). See Figure 5 Combined line chart.

4. On a per-week basis
    a. *Comparing previous week:* consumption of previous week, displayed as above
    b. *Comparing more weeks:* consumption of previous weeks, displayed as above

2.6.2 Interaction

In case of this fragment an interaction from the user is required to display the chart of a specific room. This should be accessible by a *TabView* or *Buttons* to select a room. This interaction may motivate the user to actively engage with the app (see Figure 3 Simple line chart and Figure 4 Comparing line chart).

Also to give more possibilities to compare different values (in case of the extended feature set 1a and 2a) a *DropDown* list should allow the user to set the comparing value (in case of 1a a list of previous daya respectively a list of previous weeks in 2a) – see Figure 4 Comparing line chart. Cases 1b and 2b are intentionally combined solutions and shouldn't offer the user to choose.

2.6.3 Mockup

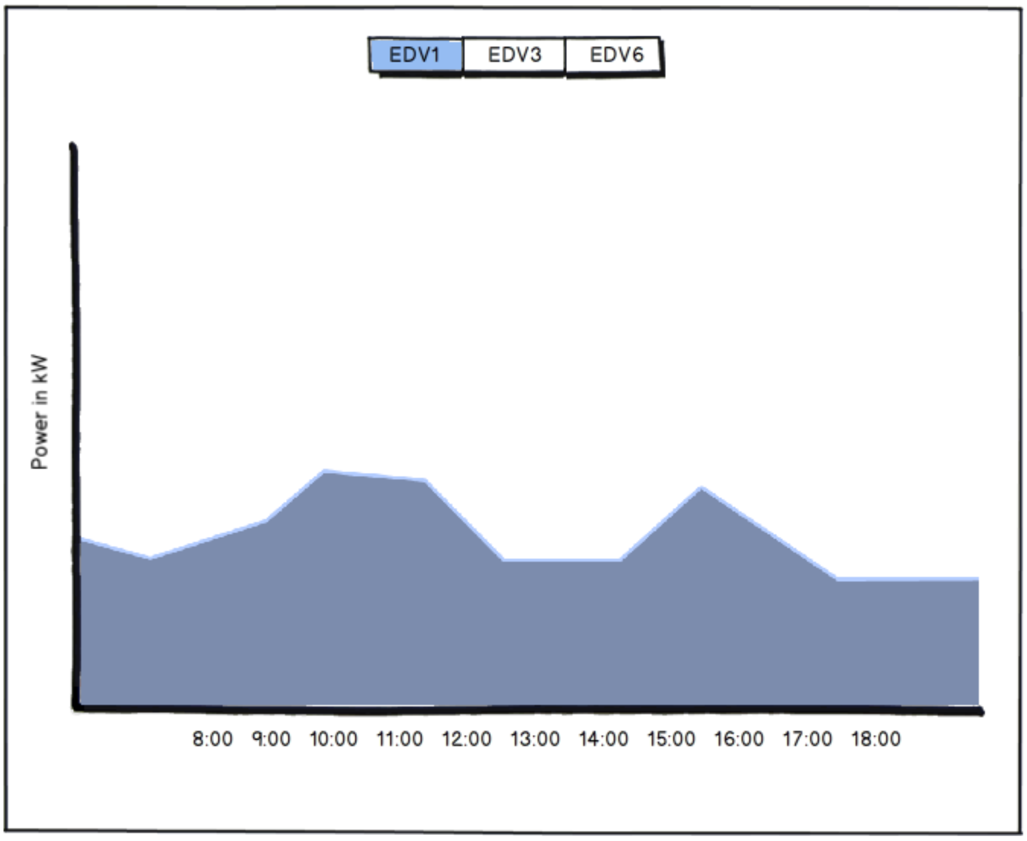

Figure 3 Simple line chart

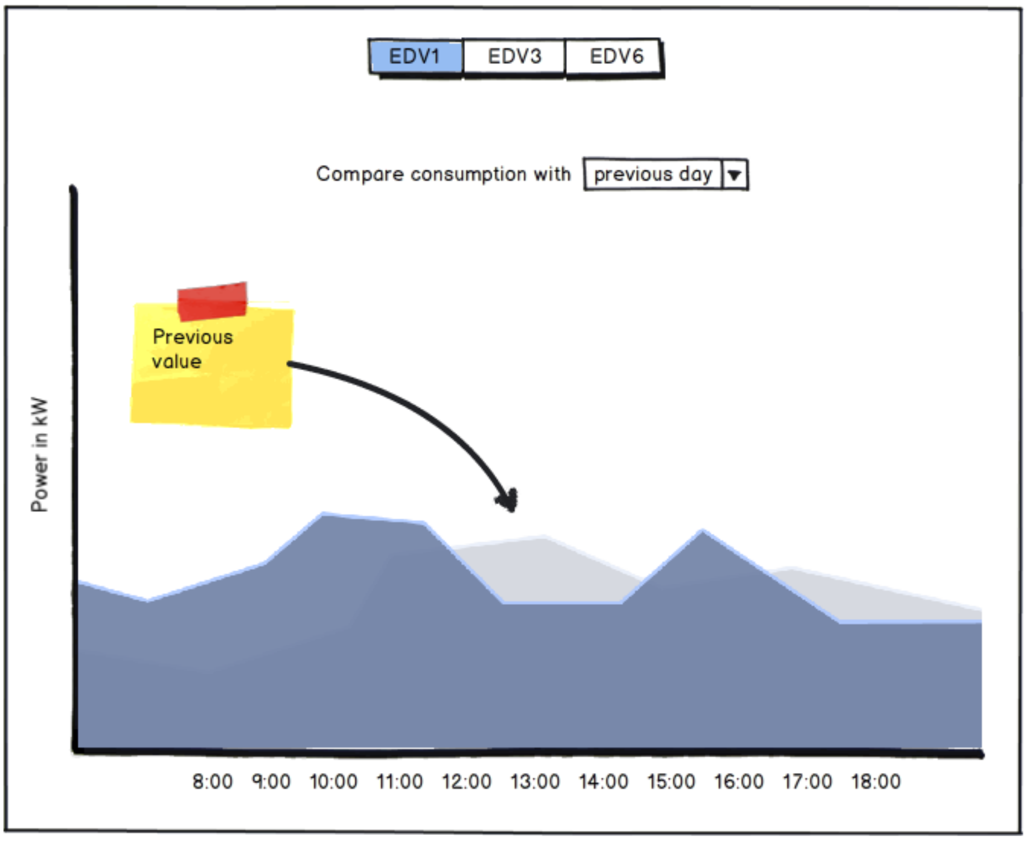

Figure 4 Comparing line chart

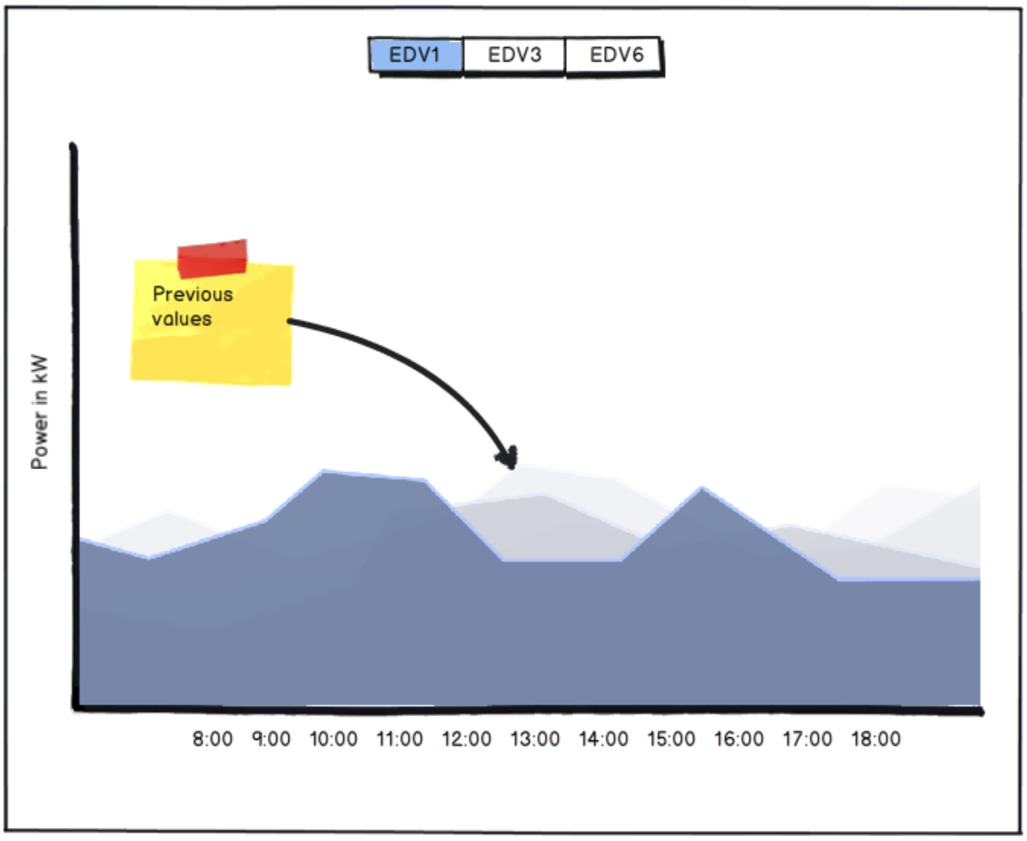

Figure 5 Combined line chart

Overall real time line chart since this indicates the behavior of this graph. Also the display of this fragment only makes sense if there's a value to compare.

2.6.4 Display parameters

1. Display type (e.g. arrow, smiley – but can be static at first)

2.6.5 Mockup

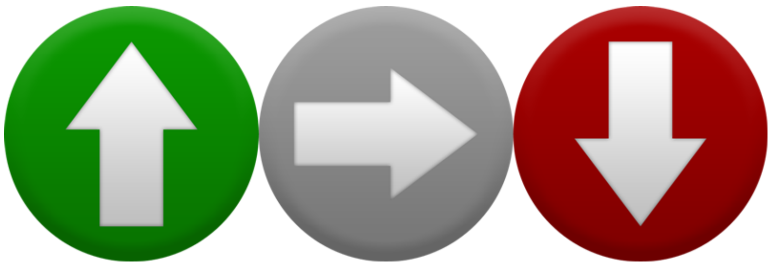

Figure 7 Success indicator

2.7 Power management service

The existing functionality of the Power Management Service should be implemented into this app. Since this feature is well described in another document (*'Power Management Service App'*) there won't be any further explanations of the features.

Still unclear, but PCs in the list should be associated with the corresponding room.

However there may be the option of a ‘light’ version (e.g. reduced possibility for remote shutdown), which is not yet defined.

2.8 Ambient signal

The purpose for this fragment is to attract attention when something is happening (or there’s a relevant notification). It is meant to work as a neutral background color but has the ability to pulsate (slow, smooth and eye-friendly transitions) in a specified color. Since a layout option could be semitransparent, this would fit very well (see Figure 8 Transparent layout: the background can remain still as nothing happens and begin to pulsate if there’s something happening).

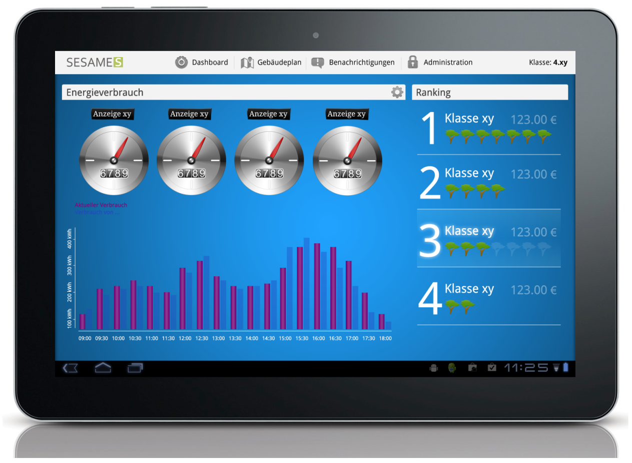

Figure 8 Transparent layout

2.8.1 Display parameters

1. A Boolean value indicating that the fragment should pulsate (or remain still otherwise)
2. Fixed base color value.

3 Layouts

Not yet decided.