

ECSP - Embedded Client Side Paradata

Stephan Schlosser¹

Center of Methods in Social Sciences, University of Göttingen

ECSP (Embedded Client Side Paradata) is a system based on HTML, JavaScript as well as CSS script, which can be implemented in the online survey software EFS-Survey/Unipark in order to capture and store client side paradata together with substantive and metadata in one database. Based on “pixel perfect” displaying of survey pages in EFS-Survey/Unipark ECSP provides the ability to capture the full paradata regardless of the used browser, as well as the used operating system. ECSP is, furthermore, capable to identify movements of the mouse cursor, mouse clicks, keystrokes, zooming, scrolling, browser resolution, as well as losing focus of the survey. All these events are provided with timestamps and are stored together with measurements of processing and transferring times.

Possible application areas of paradata in online surveys

The term “paradata” was decisively characterized by Mick Couper in 1998 during the joint Statistical meetings in Dallas in connection with survey research as “by-product of computer-assisted data collection” (Kreuter 2013, p. 2). In the literature no uniform definition is found for paradata, partially these are circumscribed with “process data” (Couper 2000; Couper and Lyberg 2005; Heerwegh 2003a; Kreuter 2013; O’Reilly 2009; Scheuren 2005). To be able to summarize more exactly the characteristics of paradata, the arising information during a data collection process can be divided in three different levels (Callegaro 2013): First, substantive data or “numeric data” (Mohler, Pennell, and Hubbard 2008) contains the direct answers of the respondents. At the second level information about data of the first level is described by metadata. Sundgren (1973, p. 116) describes this information as “data about data”, and without that information an interpretation of substantive data would not be exhaustively possible. With regard to collected data Blank and Rasmussen (2004) give the codebook as an example, which contains information about variable names and the codes of the individual values, as well as the survey field time and the recruitment basis of the study. On the third level, paradata can be distinguished from the other two levels as information describing the process of the actual data collection (Couper 2000). This information occurs while filling in a survey in (almost) real time and documents the respondent’s behavior during the interaction with the survey (Couper, Kreuter, and Lyberg 2010; Kaczmirek 2009). For instance, the information can be time measurements or the recording of information from the input devices attached to the used device, which can be collected in a non-reactive way and without an additional effort of the respondents (Kaczmirek 2009). Paradata can help to facilitate the creation and interpretation of the survey data for researchers and, where appropriate, to better assess the data quality.

¹ Center of Methods in Social Sciences, University of Göttingen, Goßlerstraße 19, 37073 Göttingen (Germany)
Email: ssschlos1@uni-goettingen.de

Server side and client side paradata

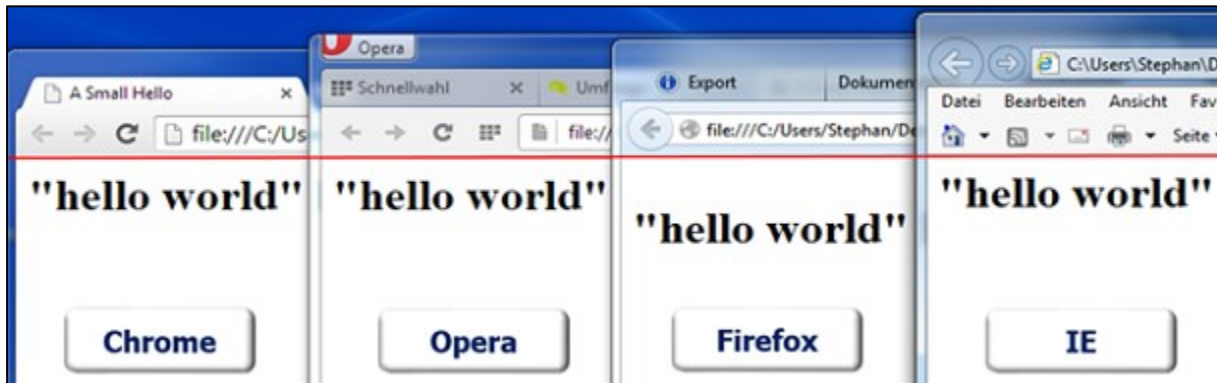
A short example shall clarify the difference between “server side” and “client side” paradata. Under a technical point of view online survey platforms use a relational database with web server support, in which the survey as well as the respondent’s answers are stored. In a synchronous data transfer between server and client, it is possible that the time between a server-side sending of a survey page and the arrival of the response is accurately detected. Within this time period, not only the encoding of the question, the retrieval and the formation of the answer and finally the delivery of the response by “clicking” at a relevant answer field takes place, but also the transport of this information to the Internet. The single time measurement therefore contains all these events. Given that this kind of paradata are technically captured exclusively at the server, Heerwegh (2003a, 2011) describes these as “servers side paradata” (abbreviated below as SSP). Parallel to SSP “client side paradata” can (abbreviated below as CSP) be recorded directly at the computer, or more precisely in the browser on the respondent’s computer. The distinction of these two paradata types is of a great importance, since on the one hand CSP can be superior in gathering complete information and on the other hand the recording of CSP requires unlike SSP, the feasibility of JavaScript or Flash in the browser (Callegaro 2013). Heerwegh describes JavaScript already in 2003 as “[...] a scripting language currently supported by almost every web browser” (p. 361). In the same year Heerwegh developed his own web-based survey platform that has the ability to capture mouse clicks to various response fields and register keystrokes on survey pages. Using recorded CSP makes it possible, once the survey page is loaded, to distinguish between the time period of filling in an answer, change the answer eventually, and sent the survey page back to the server. Unlike SSP, by using CSP it is possible to document the user activity between receiving and sending a survey page.

What is ECSP and how does it work?

ESCP differs from other script-based systems for capturing paradata in online surveys e.g. CSP (Heerwegh 2003b) or UCSP (Kaczmirek 2005) not only by its integration into EFS-Survey/Unipark, but also by a different approach of capturing the paradata. ESCP is based on the information of input devices (mouse and keyboard) which are attached to the respondent’s computer.

This approach offers the advantage to be able to capture paradata without the need to define fixed events, such as a mouse click in a particular answer field. In this way certain events can be assigned with the help of the X and Y coordinates of the mouse cursor to single survey page elements like radio-buttons or text fields (similar to the approach of eye tracking). An across different operation systems standardized survey visualization is a compelling condition for the usability of this method. For example, the use of a different browser can already lead to a different visualization of a web page. A slightly different interpretation of the HTML code represents the reason for the different visualization of the document and a dissimilar positioning of elements (see Figure 1).

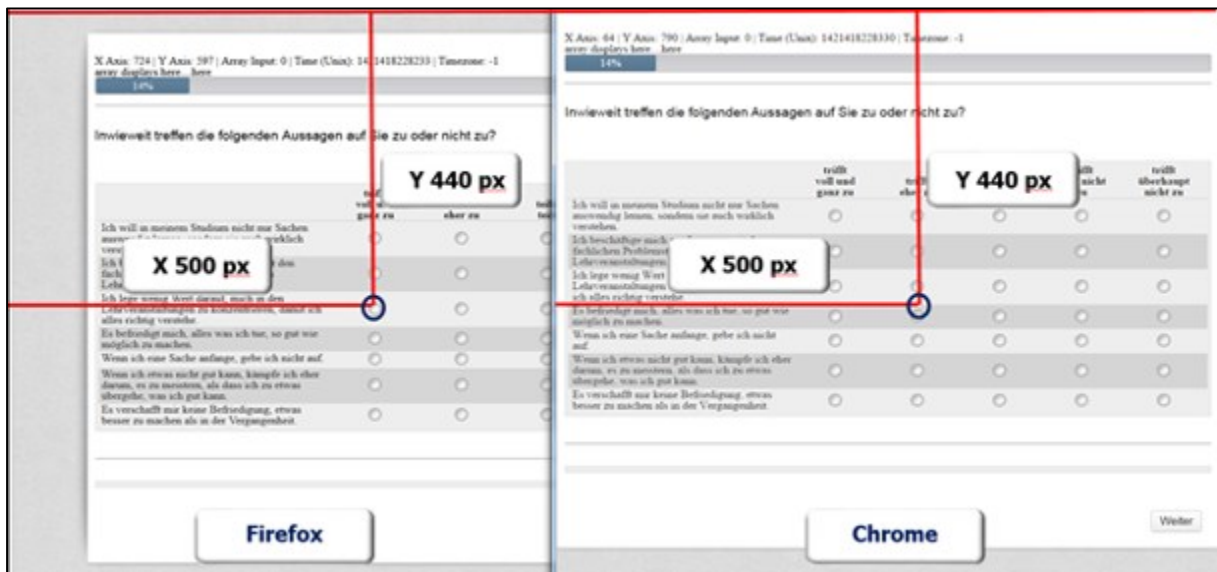
Figure 1: Visualization of a simple HTML code in different common web browsers



Despite the impactful efforts of the “World Wide Web Consortium” (W3C) to standardize the techniques that are used on the World Wide Web, the standards are not always observed under field conditions. Not all specifications – specifically ones of CSS – are binding (Resig and Bibault 2013).

The same coordinates of the mouse cursor can therefore correspond to different areas of the survey page or the same coordinates must not necessarily describe the same survey element (see Figure 2).

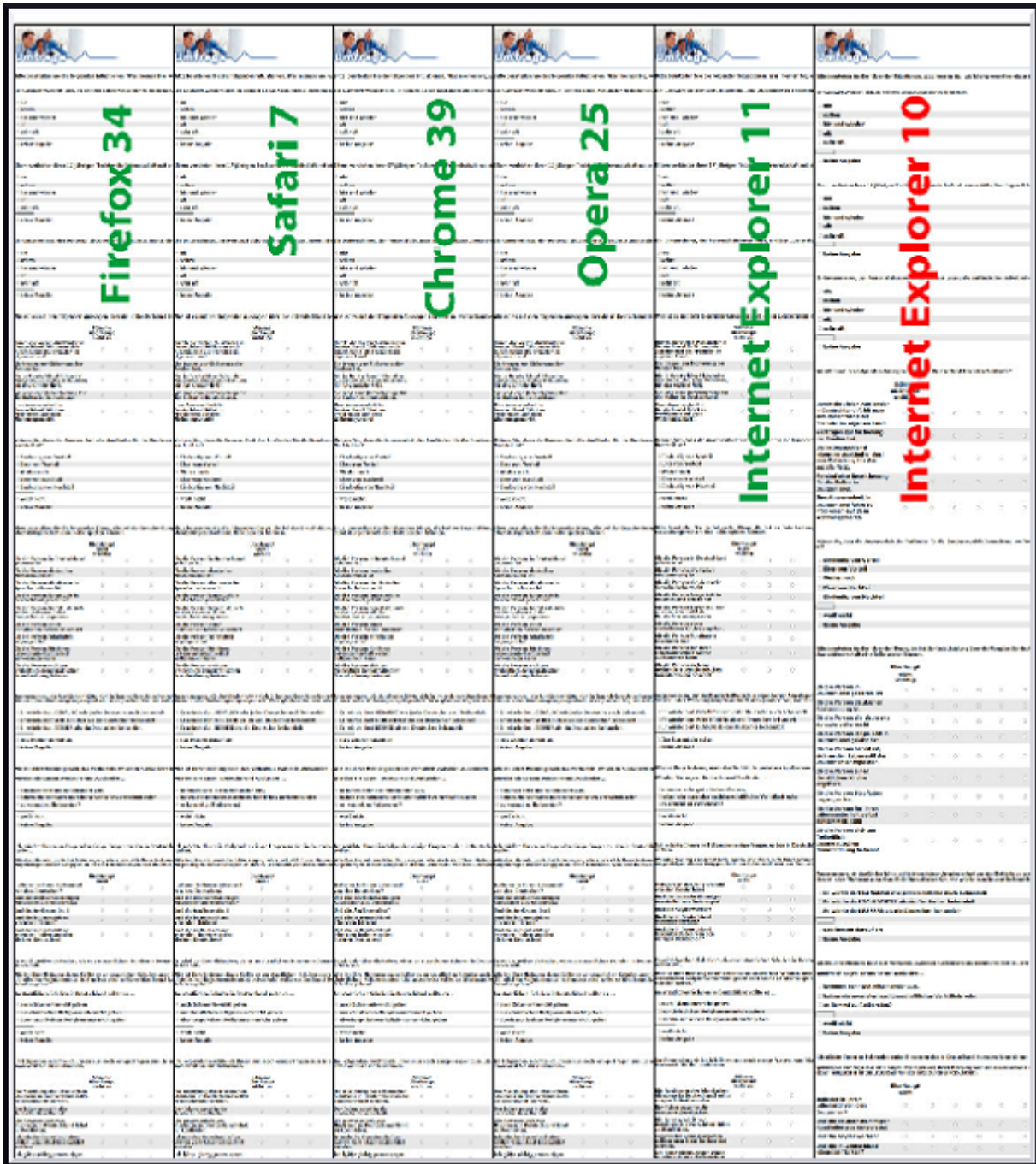
Figure 2: Visualization of coordinates in different web browsers



A possible solution for this mismatch is to account for different browser characteristics, a reset of settings and/or the normalization of browser peculiarities to a common standard (Resig and Bibault 2013). (These measures don't change the settings of the browser permanently and are only valid for a single web page with an integrated script. Respondents neither have to set cookies, nor must they load additional software from the Internet.) A combination of these measures, which is preceded the generation of survey pages by EFS-Survey/Unipark, allows the creation of pages with the exact same visualization regardless of used browser and operating system. This has been checked for the following most common browsers and different operating systems (last update: 11.12.2015): Firefox (from version 10 onward), Chrome (from version 34

onward), Safari (from version 5.1 onward), Internet Explorer (from version 11 onward), Opera (from version 20 onward), and Microsoft Edge (see Figure 3).

Figure 3: Visualization of a survey page with different common web browsers



ECSP can ensure a browser independent cross-platform to compare recordings of mouse movements, mouse clicks, keystrokes, zoom, scrolling, browser resolution, as well as losing the focus of the survey between different respondents.

Ethical considerations for using ECSP

The ethical considerations made by Heerwegh (2003b) for capturing paradata are valid also for the use of ECSP: “Use CSP only for genuine (methodological) research needs. Do not use CSP simply to “spy” on your respondents, and never use the information from CSP to replace the final answers given by the respondent on the web survey”. Additional guidelines specifically for handling paradata can be found in “Global Guideline for Online Research” (ESOMAR 2015) in Section 7.10 and 7.11. Furthermore, it is your duty to inform the respondents of the capturing of the paradata (see Code of Ethics of the German Sociological Association (DGS) and the Professional Association of German sociologists (BDS) Section 2 (3)). There is no agreement yet on a uniform form informing the respondents about the use of paradata. Also Cooper and Singer (2013), who carried out a vignette experiment with different forms of clarification about the use of paradata come to no unequivocal recommendation in this point.

How can ECSP be integrated into an EFS-Survey/Unipark project?

In this section it is explained in five steps how to integrate ECSP into an EFS-Survey/Unipark project.

It should be clearly pointed out at this juncture that the use of ESCP is at your own responsibility. Neither the author nor the institute can be held responsible for possible malfunctions of ECSP, even if ECSP should be a direct trigger for these difficulties. It should be noted that detailed pretests are strongly recommended prior the use of ECSP in the field. It is not allowed to sell this software, or to distribute it in your name. But feel free to use, copy, modify, distribute or display ESCP, as long as reference is made to the original work. The use of ECSP or individual parts of the script includes an acceptance of these conditions.

Feel free to point out errors or possible improvements of ECSP. Of course, you are mentioned as an author of any modifications.

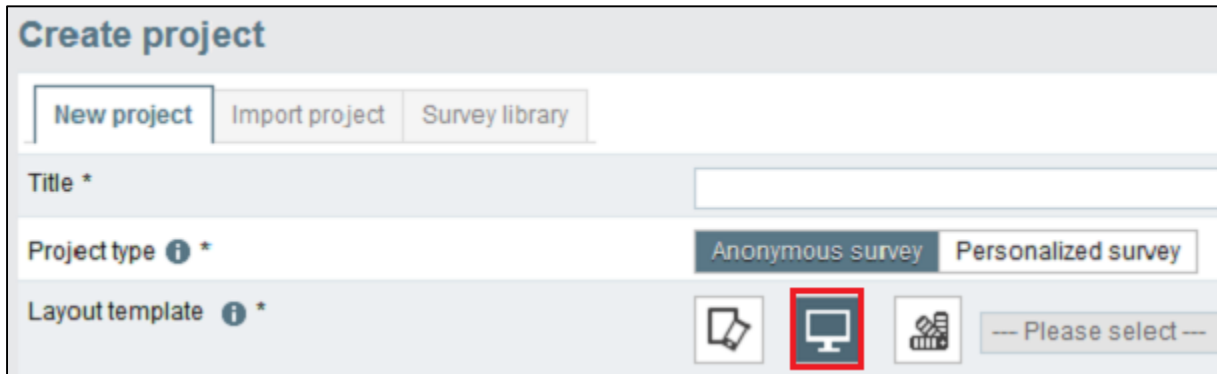
Step 1 – Preparations

Requirement: An existing project including a survey. In the most favorable case a tested and fully revised survey exists and no further changes need to be performed. Nevertheless, this does not mean that after the integration of ECSP the possibilities are limited for a change of the project. However, this action can serve to reduce possible sources of error and, for example, to hold the automated naming of variables uniform.

The survey must be provided in the “classical layout”! (This setting cannot be changed afterwards. If a survey was created in “adaptive layout”, the whole survey can be copied in a new project with “classical layout”. See Figure 4)

[Survey Menu]

Figure 4: Setting up “classical layout”

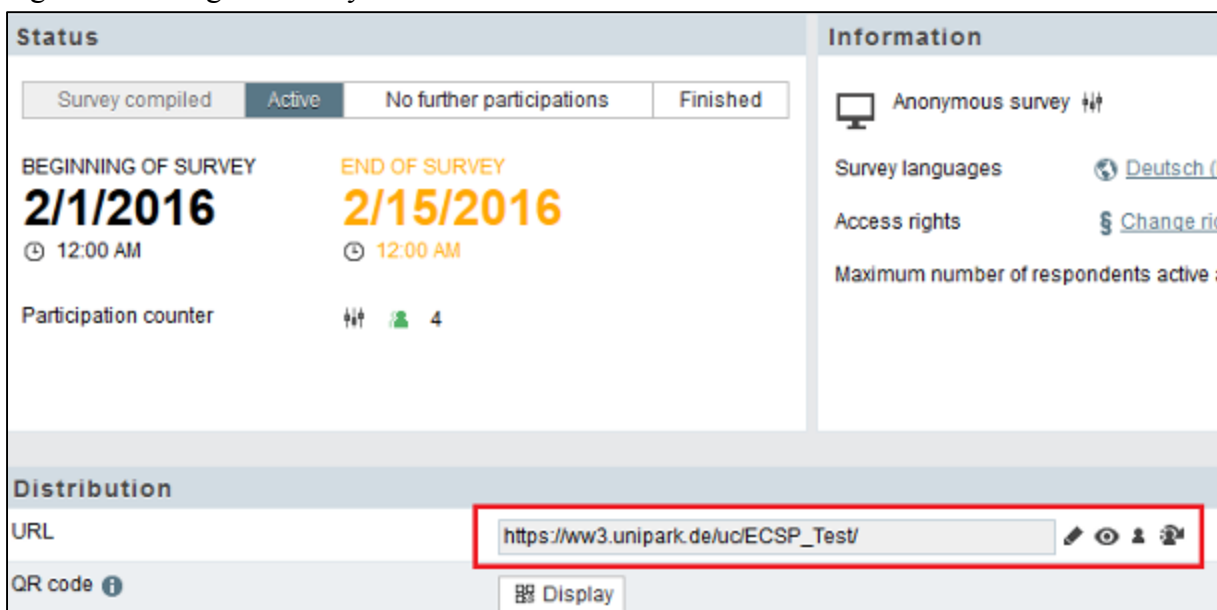


Furthermore, the web address (URL) should be adapted.

[Survey Menu]

This address is required for the integration of ECSP in a later step and should not be changed any more (see Figure 5).

Figure 5: Editing the survey address



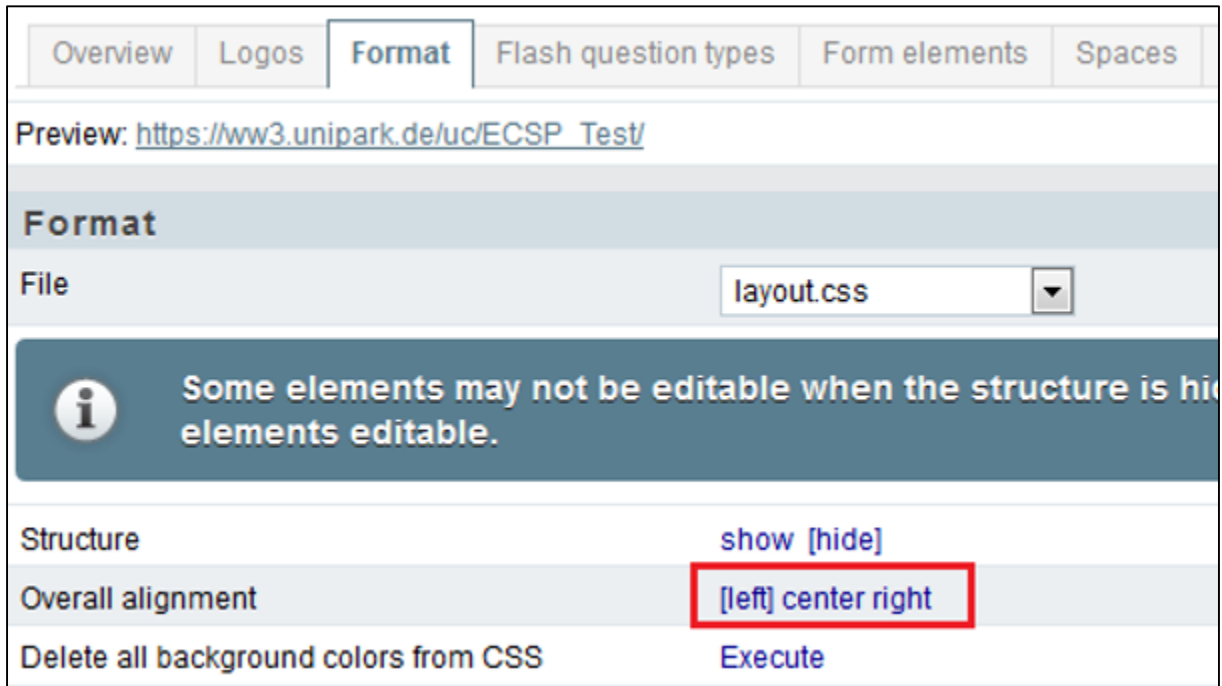
Step 2 - Normalization of the survey layout

To ensure that the survey is presented “pixel perfect” on different operating systems and in different browsers, some changes in the layout settings and in particular in the CSS default settings are necessary (see Figure 6).

The orientation of the survey layout must be set to [left].

[Survey Menu → Layout → Format] [left]

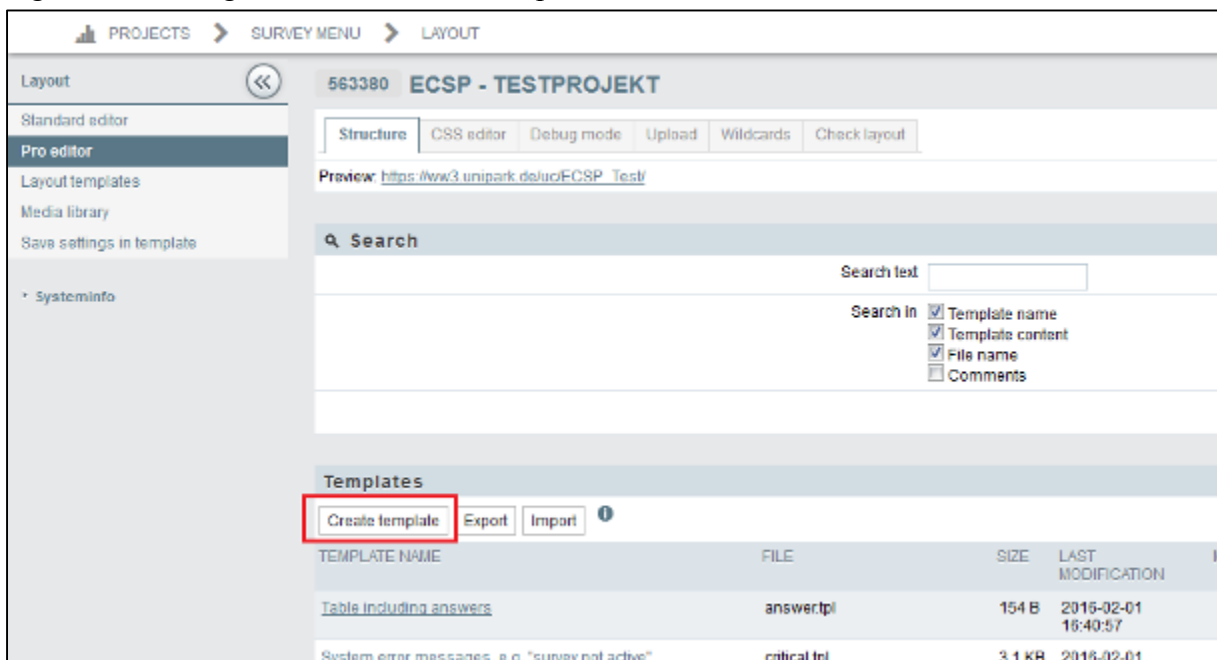
Figure 6: Editing orientation of the survey pages



In the next step, an additional CSS-template is required. This can be done by using the button [Create template].

[Survey Menu → Layout → Pro editor → Structure]

Figure 7: Creating an additional CSS-template



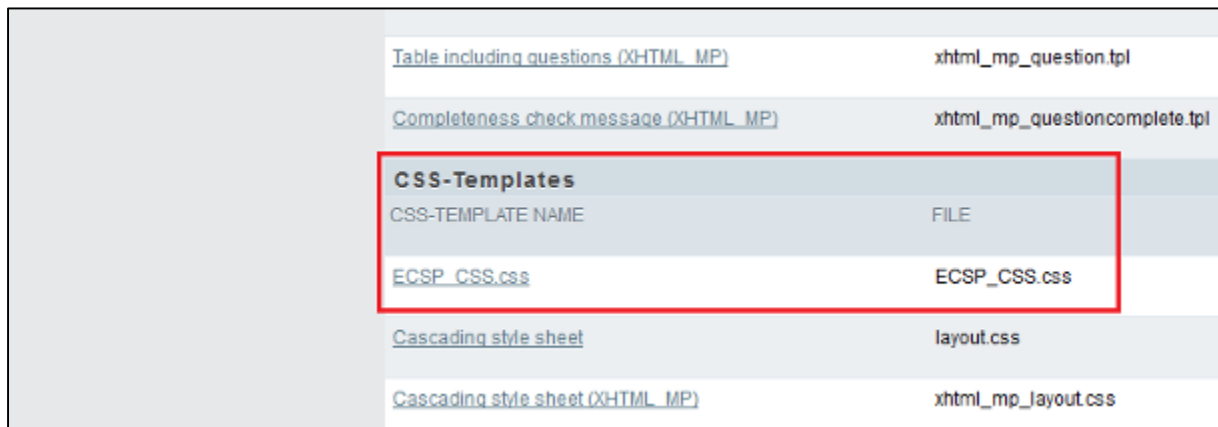
The new CSS-template should be labeled as "ECSP_CSS" and receive the file name "ECSP_CSS.css" (see Figure 8).

Figure 8: Labeling the additional CSS-template



At the end of the template list the new template is ready for further editing (see Figure 9).

Figure 9: Additional CSS-template



The following program code should be inserted in the empty template. Afterwards, the program code should be stored by clicking [Save]:

```

/* user css file */
/* http://meyerweb.com/eric/tools/css/reset/
v2.0 | 20110126
License: none (public domain)
*/
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
margin: 0;
padding: 0;
border: 0;
font-size: 100%;
font: inherit;
vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,

```



```

footer, header, hgroup, menu, nav, section {
display: block;
}
body {
line-height: 1;
}
ol, ul {
list-style: none;
}
blockquote, q {
quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
content: '';
content: none;
}
table {
border-collapse: collapse;
border-spacing: 0;
}
.input_tiny {
width: 50px;
}
.input_small {
width: 100px;
}
.input_medium {
width: 150px;
}
.input_large {
width: 200px;
}
.input_xlarge {
width: 250px;
}
.input_xxlarge {
width: 300px;
}
.input_full {
width: 100%;
}
.input_full_wrap {
display: block;
padding-right: 8px;
}
input[type="search"]::-webkit-search-decoration {
display: none;
}
input:invalid,
button:invalid,
a.button:invalid,
select:invalid,
textarea:invalid {
-webkit-box-shadow: none;
-moz-box-shadow: none;
box-shadow: none;
}
input:focus,
button:focus,
a.button:focus,
select:focus,
textarea:focus {
-webkit-box-shadow: #0066ff 0 0 5px 0;
-moz-box-shadow: #0066ff 0 0 5px 0;
box-shadow: #0066ff 0 0 5px 0;
z-index: 1;
}
input[type="file"]:focus, input[type="file"]:active,
input[type="radio"]:focus,
input[type="radio"]:active,
input[type="checkbox"]:focus,
input[type="checkbox"]:active {
-webkit-box-shadow: none;
-moz-box-shadow: none;
box-shadow: none;
}

```

```

}
button,
a.button,
input[type="reset"],
input[type="submit"],
input[type="button"] {
  -webkit-appearance: none;
  -webkit-border-radius: 4px;
  -moz-border-radius: 4px;
  -ms-border-radius: 4px;
  -o-border-radius: 4px;
  border-radius: 4px;
  -webkit-background-clip: padding;
  -moz-background-clip: padding;
  background-clip: padding-box;
  background: #dddddd url('../images/button.png?1298351022') repeat-x;
  background-image: -webkit-gradient(linear, 50% 0%, 50% 100%, color-stop(0%, #ffffff), color-stop(100%, #dddddd));
  background-image: -moz-linear-gradient(#ffffff, #dddddd);
  background-image: -o-linear-gradient(#ffffff, #dddddd);
  background-image: linear-gradient(#ffffff, #dddddd);
  border: 1px solid;
  border-color: #dddddd #bbbbbb #999999;
  cursor: pointer;
  color: #333333;
  display: inline-block;
  font: bold 12px/1.3 "Helvetica Neue", Arial, "Liberation Sans", FreeSans, sans-serif;
  outline: 0;
  overflow: visible;
  margin: 0;
  padding: 3px 10px;
  text-shadow: white 0 1px 1px;
  text-decoration: none;
  vertical-align: top;
  width: auto;
  *padding-top: 2px;
  *padding-bottom: 0;
}
button:hover,
a.button:hover,
input[type="reset"]:hover,
input[type="submit"]:hover,
input[type="button"]:hover {
  background-image: -webkit-gradient(linear, 50% 0%, 50% 100%, color-stop(0%, #ffffff), color-stop(1px, #eeeeee), color-stop(100%, #cccccc));
  background-image: -webkit-linear-gradient(#ffffff, #eeeeee 1px, #cccccc);
  background-image: -moz-linear-gradient(#ffffff, #eeeeee 1px, #cccccc);
  background-image: -o-linear-gradient(#ffffff, #eeeeee 1px, #cccccc);
  background-image: linear-gradient(#ffffff, #eeeeee 1px, #cccccc);
  text-decoration: none;
}
button:active,
a.button:active,
input[type="reset"]:active,
input[type="submit"]:active,
input[type="button"]:active {
  background-image: -webkit-gradient(linear, 50% 0%, 50% 100%, color-stop(0%, #dddddd), color-stop(100%, #eeeeee));
  background-image: -webkit-linear-gradient(#dddddd, #eeeeee);
  background-image: -moz-linear-gradient(#dddddd, #eeeeee);
  background-image: -o-linear-gradient(#dddddd, #eeeeee);
  background-image: linear-gradient(#dddddd, #eeeeee);
  -webkit-box-shadow: inset rgba(0, 0, 0, 0.25) 0 1px 2px 0;
  -moz-box-shadow: inset rgba(0, 0, 0, 0.25) 0 1px 2px 0;
  box-shadow: inset rgba(0, 0, 0, 0.25) 0 1px 2px 0;
  border-color: #999999 #bbbbbb #dddddd;
}
button::-moz-focus-inner,
a.button::-moz-focus-inner,
input[type="reset"]::-moz-focus-inner,
input[type="submit"]::-moz-focus-inner,
input[type="button"]::-moz-focus-inner {
  border: 0;
  padding: 0;
}

```

```

a.button {
  *padding-bottom: 3px;
}
button {
  *padding-top: 1px;
  *padding-bottom: 1px;
}
textarea,
select,
input[type="date"],
input[type="datetime"],
input[type="datetime-local"],
input[type="email"],
input[type="month"],
input[type="number"],
input[type="password"],
input[type="search"],
input[type="tel"],
input[type="text"],
input[type="time"],
input[type="url"],
input[type="week"] {
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
  -webkit-background-clip: padding;
  -moz-background-clip: padding;
  background-clip: padding-box;
  -webkit-border-radius: 0;
  -moz-border-radius: 0;
  -ms-border-radius: 0;
  -o-border-radius: 0;
  border-radius: 0;
  -webkit-appearance: none;
  background-color: white;
  border: 1px solid;
  border-color: #848484 #c1c1c1 #e1e1e1;
  color: black;
  outline: 0;
  margin: 0;
  padding: 2px 3px;
  text-align: left;
  font-size: 13px;
  font-family: Arial, "Liberation Sans", FreeSans, sans-serif;
  height: 1.8em;
  vertical-align: top;
  *padding-top: 2px;
  *padding-bottom: 1px;
  *height: auto;
}
textarea[disabled],
select[disabled],
input[type="date"][disabled],
input[type="datetime"][disabled],
input[type="datetime-local"][disabled],
input[type="email"][disabled],
input[type="month"][disabled],
input[type="number"][disabled],
input[type="password"][disabled],
input[type="search"][disabled],
input[type="tel"][disabled],
input[type="text"][disabled],
input[type="time"][disabled],
input[type="url"][disabled],
input[type="week"][disabled] {
  background-color: #eeeeee;
}
button[disabled],
input[disabled],
select[disabled],
select[disabled] option,
select[disabled] optgroup,
textarea[disabled],
a.button_disabled {
  -webkit-box-shadow: none;
  -moz-box-shadow: none;
}

```

```

    box-shadow: none;
    -moz-user-select: -moz-none;
    -webkit-user-select: none;
    -khtml-user-select: none;
    user-select: none;
    color: #888888;
    cursor: default;
}
input::-webkit-input-placeholder,
textarea::-webkit-input-placeholder {
    color: #888888;
}
input:-moz-placeholder,
textarea:-moz-placeholder {
    color: #888888;
}
input.placeholder_text,
textarea.placeholder_text {
    color: #888888;
}
textarea,
select[size],
select[multiple] {
    height: auto;
}
select[size="0"],
select[size="1"] {
    height: 1.8em;
    *height: auto;
}
@media (-webkit-min-device-pixel-ratio: 0) {
    select[size],
    select[multiple],
    select[multiple][size] {
        background-image: none;
        padding-right: 3px;
    }

    select,
    select[size="0"],
    select[size="1"] {
        background-image:
url(data:image/png;base64,R0lGODlhDQAEIAAAAAAAAAAP8A/yH5BAEHAAEALAAAAAAAAAAQAAAIhA+hG5jMDpxvhgIAOw==);
        background-repeat: no-repeat;
        background-position: right center;
        padding-right: 20px;
    }

    ::-webkit-validation-bubble-message {
        -webkit-box-shadow: none;
        box-shadow: none;
        background: -webkit-gradient(linear, left top, left bottom, color-stop(0, #666666), color-stop(1,
black));
        border: 0;
        color: white;
        font: 13px/17px "Lucida Grande", Arial, "Liberation Sans", FreeSans, sans-serif;
        overflow: hidden;
        padding: 15px 15px 17px;
        text-shadow: black 0 0 1px;
        min-height: 16px;
    }

    ::-webkit-validation-bubble-arrow,
    ::-webkit-validation-bubble-top-outer-arrow,
    ::-webkit-validation-bubble-top-inner-arrow {
        -webkit-box-shadow: none;
        box-shadow: none;
        background: #666666;
        border-color: #666666;
    }
}
textarea {
    min-height: 40px;
    overflow: auto;
    resize: vertical;
    width: 100%;
}

```

```

}
optgroup {
  color: black;
  font-style: normal;
  font-weight: normal;
  font-family: Arial, "Liberation Sans", FreeSans, sans-serif;
}
optgroup::-moz-focus-inner {
  border: 0;
  padding: 0;
}
.ie6_button,
* html button,
* html a.button {
  background: #dddddd url('../images/button.png?1298351022') repeat-x;
  border: 1px solid;
  border-color: #dddddd #bbbbbb #999999;
  cursor: pointer;
  color: #333333;
  font: bold 12px/1.2 Arial, sans-serif;
  padding: 2px 10px 0;
  text-decoration: none;
  overflow: visible;
  vertical-align: top;
  width: auto;
}
* html a.button {
  position: relative;
  top: 3px;
  padding-bottom: 2px;
}
* html button {
  padding-top: 1px;
  padding-bottom: 1px;
}
.ie6_input,
* html textarea,
* html select {
  background: white;
  border: 1px solid;
  border-color: #848484 #c1c1c1 #e1e1e1;
  color: black;
  padding: 2px 3px 1px;
  font-size: 13px;
  font-family: Arial, sans-serif;
  vertical-align: top;
}

* html select {
  margin-top: 1px;
}
.placeholder_text,
.ie6_input_disabled,
.ie6_button_disabled {
  color: #888888;
}
.ie6_input_disabled {
  background: #eeeeee;
}
/*! normalize.css v3.0.2 | MIT License | git.io/normalize */
/*!Copyright (c) Nicolas Gallagher and Jonathan Neal */
/*!Permission is hereby granted, free of charge, to any person obtaining a copy of this software and
associated documentation files
*(the "Software"), to deal in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, */
/*!publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom
the Software is furnished to do so, */
/*!subject to the following conditions: */
/*!The above copyright notice and this permission notice shall be included in all copies or substantial
portions of the Software. */
/**
 * 1. Set default font family to sans-serif.
 * 2. Prevent iOS text size adjust after orientation change, without disabling
 * user zoom.
 */
html {

```

```

font-family: sans-serif; /* 1 */
-ms-text-size-adjust: 100%; /* 2 */
-webkit-text-size-adjust: 100%; /* 2 */
}
/**
 * Remove default margin.
 */
body {
margin: 0;
}

/* HTML5 display definitions
===== */
/**
 * Correct `block` display not defined for any HTML5 element in IE 8/9.
 * Correct `block` display not defined for `details` or `summary` in IE 10/11
 * and Firefox.
 * Correct `block` display not defined for `main` in IE 11.
 */
article,
aside,
details,
figcaption,
figure,
footer,
header,
hgroup,
main,
menu,
nav,
section,
summary {
display: block;
}
/**
 * 1. Correct `inline-block` display not defined in IE 8/9.
 * 2. Normalize vertical alignment of `progress` in Chrome, Firefox, and Opera.
 */
audio,
canvas,
progress,
video {
display: inline-block; /* 1 */
vertical-align: baseline; /* 2 */
}
/**
 * Prevent modern browsers from displaying `audio` without controls.
 * Remove excess height in iOS 5 devices.
 */
audio:not([controls]) {
display: none;
height: 0;
}
/**
 * Address `[hidden]` styling not present in IE 8/9/10.
 * Hide the `template` element in IE 8/9/11, Safari, and Firefox < 22.
 */
[hidden],
template {
display: none;
}

/* Links
===== */
/**
 * Remove the gray background color from active links in IE 10.
 */
a {
background-color: transparent;
}
/**
 * Improve readability when focused and also mouse hovered in all browsers.
 */
a:active,

```

```

a:hover {
    outline: 0;
}

/* Text-level semantics
===== */
/**
 * Address styling not present in IE 8/9/10/11, Safari, and Chrome.
 */

abbr[title] {
    border-bottom: 1px dotted;
}
/**
 * Address style set to `bolder` in Firefox 4+, Safari, and Chrome.
 */
b,
strong {
    font-weight: bold;
}

/**
 * Address styling not present in Safari and Chrome.
 */
dfn {
    font-style: italic;
}
/**
 * Address variable `h1` font-size and margin within `section` and `article`
 * contexts in Firefox 4+, Safari, and Chrome.
 */
h1 {
    font-size: 2em;
    margin: 0.67em 0;
}
/**
 * Address styling not present in IE 8/9.
 */
mark {
    background: #ff0;
    color: #000;
}
/**
 * Address inconsistent and variable font size in all browsers.
 */
small {
    font-size: 80%;
}
/**
 * Prevent `sub` and `sup` affecting `line-height` in all browsers.
 */

sub,
sup {
    font-size: 75%;
    line-height: 0;
    position: relative;
    vertical-align: baseline;
}
sup {
    top: -0.5em;
}
sub {
    bottom: -0.25em;
}

/* Embedded content
===== */
/**
 * Remove border when inside `a` element in IE 8/9/10.
 */
img {
    border: 0;
}

```

```

/**
 * Correct overflow not hidden in IE 9/10/11.
 */
svg:not(:root) {
  overflow: hidden;
}

/* Grouping content
===== */
/**
 * Address margin not present in IE 8/9 and Safari.
 */
figure {
  margin: 1em 40px;
}
/**
 * Address differences between Firefox and other browsers.
 */
hr {
  -moz-box-sizing: content-box;
  box-sizing: content-box;
  height: 0;
}
/**
 * Contain overflow in all browsers.
 */
pre {
  overflow: auto;
}
/**
 * Address odd `em`-unit font size rendering in all browsers.
 */
code,
kbd,
pre,
samp {
  font-family: monospace, monospace;
  font-size: 1em;
}

/* Forms
===== */
/**
 * Known limitation: by default, Chrome and Safari on OS X allow very limited
 * styling of `select`, unless a `border` property is set.
 */
/**
 * 1. Correct color not being inherited.
 *    Known issue: affects color of disabled elements.
 * 2. Correct font properties not being inherited.
 * 3. Address margins set differently in Firefox 4+, Safari, and Chrome.
 */
button,
input,
optgroup,
select,
textarea {
  color: inherit; /* 1 */
  font: inherit; /* 2 */
  margin: 0; /* 3 */
}
/**
 * Address `overflow` set to `hidden` in IE 8/9/10/11.
 */
button {
  overflow: visible;
}
/**
 * Address inconsistent `text-transform` inheritance for `button` and `select`.
 * All other form control elements do not inherit `text-transform` values.
 * Correct `button` style inheritance in Firefox, IE 8/9/10/11, and Opera.
 * Correct `select` style inheritance in Firefox.
 */
button,

```



```

select {
  text-transform: none;
}
/**
 * 1. Avoid the WebKit bug in Android 4.0.* where (2) destroys native `audio`
 *    and `video` controls.
 * 2. Correct inability to style clickable `input` types in iOS.
 * 3. Improve usability and consistency of cursor style between image-type
 *    `input` and others.
 */
button,
html input[type="button"], /* 1 */
input[type="reset"],
input[type="submit"] {
  -webkit-appearance: button; /* 2 */
  cursor: pointer; /* 3 */
}
/**
 * Re-set default cursor for disabled elements.
 */
button[disabled],
html input[disabled] {
  cursor: default;
}
/**
 * Remove inner padding and border in Firefox 4+.
 */
button::-moz-focus-inner,
input::-moz-focus-inner {
  border: 0;
  padding: 0;
}
/**
 * Address Firefox 4+ setting `line-height` on `input` using `!important` in
 * the UA stylesheet.
 */
input {
  line-height: normal;
}
/**
 * It's recommended that you don't attempt to style these elements.
 * Firefox's implementation doesn't respect box-sizing, padding, or width.
 *
 * 1. Address box sizing set to `content-box` in IE 8/9/10.
 * 2. Remove excess padding in IE 8/9/10.
 */
input[type="checkbox"],
input[type="radio"] {
  box-sizing: border-box; /* 1 */
  padding: 0; /* 2 */
}
/**
 * Fix the cursor style for Chrome's increment/decrement buttons. For certain
 * `font-size` values of the `input`, it causes the cursor style of the
 * decrement button to change from `default` to `text`.
 */
input[type="number"]::-webkit-inner-spin-button,
input[type="number"]::-webkit-outer-spin-button {
  height: auto;
}
/**
 * 1. Address `appearance` set to `searchfield` in Safari and Chrome.
 * 2. Address `box-sizing` set to `border-box` in Safari and Chrome
 *    (include `-moz` to future-proof).
 */
input[type="search"] {
  -webkit-appearance: textfield; /* 1 */
  -moz-box-sizing: content-box;
  -webkit-box-sizing: content-box; /* 2 */
  box-sizing: content-box;
}
/**
 * Remove inner padding and search cancel button in Safari and Chrome on OS X.
 * Safari (but not Chrome) clips the cancel button when the search input has
 * padding (and `textfield` appearance).

```

```

*/
input[type="search"]::-webkit-search-cancel-button,
input[type="search"]::-webkit-search-decoration {
  -webkit-appearance: none;
}
/**
 * Define consistent border, margin, and padding.
 */
fieldset {
  border: 1px solid #c0c0c0;
  margin: 0 2px;
  padding: 0.35em 0.625em 0.75em;
}
/**
 * 1. Correct `color` not being inherited in IE 8/9/10/11.
 * 2. Remove padding so people aren't caught out if they zero out fieldsets.
 */
legend {
  border: 0; /* 1 */
  padding: 0; /* 2 */
}
/**
 * Remove default vertical scrollbar in IE 8/9/10/11.
 */
textarea {
  overflow: auto;
}
/**
 * Don't inherit the `font-weight` (applied by a rule above).
 * NOTE: the default cannot safely be changed in Chrome and Safari on OS X.
 */
optgroup {
  font-weight: bold;
}

/* Tables
===== */
/**
 * Remove most spacing between table cells.
 */
table {
  border-collapse: collapse;
  border-spacing: 0;
}
td,
th {
  padding: 0;
}
/* 50px to the right */
BODY { margin-left: 50px; }

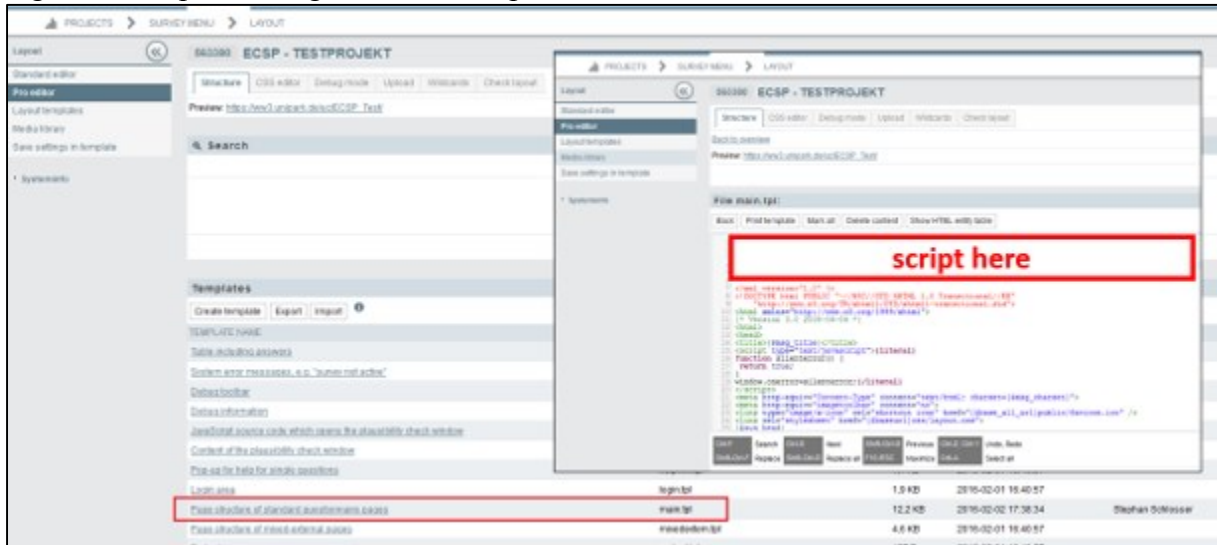
```

The code contains two free of license CSS scripts for a “CSS-reset” and a “normalization” of the CSS default settings (“RESET CSS 2.0” by Eric A. and Kathryn S. Meyer <http://meyerweb.com/eric/tools/css/reset/> “normalize.css 3.0.2” by Nicolas Gallagher and Jonathan Neal <https://necolas.github.io/normalize.css/>).

Step 3 - Implementing the ECSP script

In the next step, the ECSP script is integrated into the already existing template “Page structure of standard questionnaire pages” [[main.tpl](#)]. The following program code should be copied in this template before or above the already existing code (see Figure 10).

Figure 10: Implementing the ECSP script



```
{* ECSP 1.0 01.02.16 *}
<html>
<head>{$sys_head}

{* JQUERY *}
<script type='text/javascript' src="//ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>

{* to be adjusted to the projekt link! *}
<link rel="stylesheet" href="//ww3.unipark.de/uc/ECSP_Test/css/ECSP_CSS.css" />

</head>
<title>{$msg_title}</title>
<script type='text/javascript'>

{literal}

jQuery.noConflict();

var now = (function() {

var performance = window.performance || {};

performance.now = (function() {
return performance.now ||
performance.webkitNow ||
performance.msNow ||
performance.oNow ||
performance.mozNow ||
function() { return new Date().getTime(); };
})();
return performance.now();
});

var start = now();

x = new Date();
var XArray = new Array();
var YArray = new Array();
var TArray = new Array();
var Xtext = '';
var Ytext = '';
var Ttext = '';
var ClickXArray = new Array();
var ClickYArray = new Array();
var ClickTArray = new Array();
var ClickXtext = '';
var ClickYtext = '';
var ClickTtext = '';

(function($) {
```

```

$(document).ready(function () {
    $(document.body).mousemove(function (e) {
        if (e.pageX !== 0 && e.pageY !== 0) {
            XArray[XArray.length] = Math.round(e.pageX);
            YArray[YArray.length] = Math.round(e.pageY);
            TArray[TArray.length] = Math.round(now() - start);
        }
    });
});
})(jQuery);

(function($) {
    $(function () {
        $(document.body).mousedown(function (f) {
            ClickXArray[ClickXArray.length] = Math.round(f.pageX);
            ClickYArray[ClickYArray.length] = Math.round(f.pageY);
            ClickTArray[ClickTArray.length] = Math.round(now() - start);
        });
    });
})(jQuery);

var SizeXArray = new Array();
var SizeYArray = new Array();
var SizeTArray = new Array();
var SizeXtext = '';
var SizeYtext = '';
var SizeTtext = '';
var ZoomArray = new Array();
var Zoomtext = '';

(function($) {
function onresize (){
    var Zoomtext = ((window.outerWidth -16) / window.innerWidth);
    SizeXArray[SizeXArray.length] = $(window).width();
    SizeYArray[SizeYArray.length] = $(window).height();
    SizeTArray[SizeTArray.length] = new Date().getTime() - x.getTime();
    ZoomArray[ZoomArray.length] = Zoomtext;
};
    $(window).resize(onresize);
onresize ();// first time
})(jQuery);

var ScrollTArray = new Array();
var ScrollYArray = new Array();
var scroll = '';

(function($) {
$(window).scroll(function () {
    var scroll = $(document).scrollTop();
    ScrollYArray[ScrollYArray.length] = scroll;
    ScrollTArray[ScrollTArray.length] = new Date().getTime() - x.getTime();
});
})(jQuery);

var KeyTArray = new Array();
var KeyCodeArray = new Array();
var event = '';

(function($) {
document.onkeypress=function(k){
    var k=window.event || k;
    KeyCodeArray[KeyCodeArray.length] = (k.which || k.keyCode);
    KeyTArray[KeyTArray.length] = new Date().getTime() - x.getTime();
};
})(jQuery);

var FocusTArray = new Array();
var FocusArray = new Array();
var state = '';
var visibilityChange = (function (window) {
    var inView = false;
    return function (fn) {
        window.onfocus = window.onblur = window.onpageshow = window.onpagehide = function (s) {
            if ({{focus:1, pageshow:1}}[s.type]) {
                if (inView) return;
                if (inView) return;
                fn("1"); //visible
            }
        };
    };
})(window);

```

```

        inView = true;
    } else if (inView) {
        fn("0"); //hidden
        inView = false;
    }
    };
};
})(this));

visibilityChange(function (state) {
    FocusArray[FocusArray.length] = state;
    FocusTArray[FocusTArray.length] = new Date().getTime() - x.getTime();
    //document.body.innerHTML += state + " Time:" + FocusTArray + "<br>";
});
function send()
{
    y = new Date();
    diff = y.getTime() - x.getTime();

var ZeitDiff = document.forms[0].varZeitDiff.value;
var ZeitDiff_storage = eval("document.forms[0]."+ZeitDiff);
ZeitDiff_storage.value = diff;

MessStart = x.getTime();
var ZeitStart = document.forms[0].varZeitStart.value;
var ZeitStart_storage = eval("document.forms[0]."+ZeitStart);
ZeitStart_storage.value = MessStart;

var XArrayOut = document.forms[0].varXArrayOut.value;
var XArrayOut_storage = eval("document.forms[0]."+XArrayOut);
XArrayOut_storage.value = XArray;

var YArrayOut = document.forms[0].varYArrayOut.value;
var YArrayOut_storage = eval("document.forms[0]."+YArrayOut);
YArrayOut_storage.value = YArray;

var TArrayOut = document.forms[0].varTArrayOut.value;
var TArrayOut_storage = eval("document.forms[0]."+TArrayOut);
TArrayOut_storage.value = TArray;

var ClickTArrayOut = document.forms[0].varClickTArrayOut.value;
var ClickTArrayOut_storage = eval("document.forms[0]."+ClickTArrayOut);
ClickTArrayOut_storage.value = ClickTArray;

var ClickXArrayOut = document.forms[0].varClickXArrayOut.value;
var ClickXArrayOut_storage = eval("document.forms[0]."+ClickXArrayOut);
ClickXArrayOut_storage.value = ClickXArray;

var ClickYArrayOut = document.forms[0].varClickYArrayOut.value;
var ClickYArrayOut_storage = eval("document.forms[0]."+ClickYArrayOut);
ClickYArrayOut_storage.value = ClickYArray;

var SizeTArrayOut = document.forms[0].varSizeTArrayOut.value;
var SizeTArrayOut_storage = eval("document.forms[0]."+SizeTArrayOut);
SizeTArrayOut_storage.value = SizeTArray;

var SizeXArrayOut = document.forms[0].varSizeXArrayOut.value;
var SizeXArrayOut_storage = eval("document.forms[0]."+SizeXArrayOut);
SizeXArrayOut_storage.value = SizeXArray;

var SizeYArrayOut = document.forms[0].varSizeYArrayOut.value;
var SizeYArrayOut_storage = eval("document.forms[0]."+SizeYArrayOut);
SizeYArrayOut_storage.value = SizeYArray;

var ScrollTArrayOut = document.forms[0].varScrollTArrayOut.value;
var ScrollTArrayOut_storage = eval("document.forms[0]."+ScrollTArrayOut);
ScrollTArrayOut_storage.value = ScrollTArray;

var ScrollYArrayOut = document.forms[0].varScrollYArrayOut.value;
var ScrollYArrayOut_storage = eval("document.forms[0]."+ScrollYArrayOut);
ScrollYArrayOut_storage.value = ScrollYArray;

var ZoomArrayOut = document.forms[0].varZoomArrayOut.value;
var ZoomArrayOut_storage = eval("document.forms[0]."+ZoomArrayOut);
ZoomArrayOut_storage.value = ZoomArray;

```

```

var KeyTArrayOut = document.forms[0].varKeyTArrayOut.value;
var KeyTArrayOut_storage = eval("document.forms[0]."+KeyTArrayOut);
KeyTArrayOut_storage.value = KeyTArray;

var KeyCodeArrayOut = document.forms[0].varKeyCodeArrayOut.value;
var KeyCodeArrayOut_storage = eval("document.forms[0]."+KeyCodeArrayOut);
KeyCodeArrayOut_storage.value = KeyCodeArray;

var FocusTArrayOut = document.forms[0].varFocusTArrayOut.value;
var FocusTArrayOut_storage = eval("document.forms[0]."+FocusTArrayOut);
FocusTArrayOut_storage.value = FocusTArray;

var FocusArrayOut = document.forms[0].varFocusArrayOut.value;
var FocusArrayOut_storage = eval("document.forms[0]."+FocusArrayOut);
FocusArrayOut_storage.value = FocusArray;
}
</script>
{/literal}
{* END ECSP *}

```

In line nine of this template “main.tpl” a linkage to the already created CSS-template must be established (ECSP_CSS). For that purpose, this program line is adjusted to the survey web address (see Figure 11 and example below).

Example: If the web address (URL) defined in step 1 would be ...

https://ww3.unipark.de/eee/questionnaire_Test/

The line ...

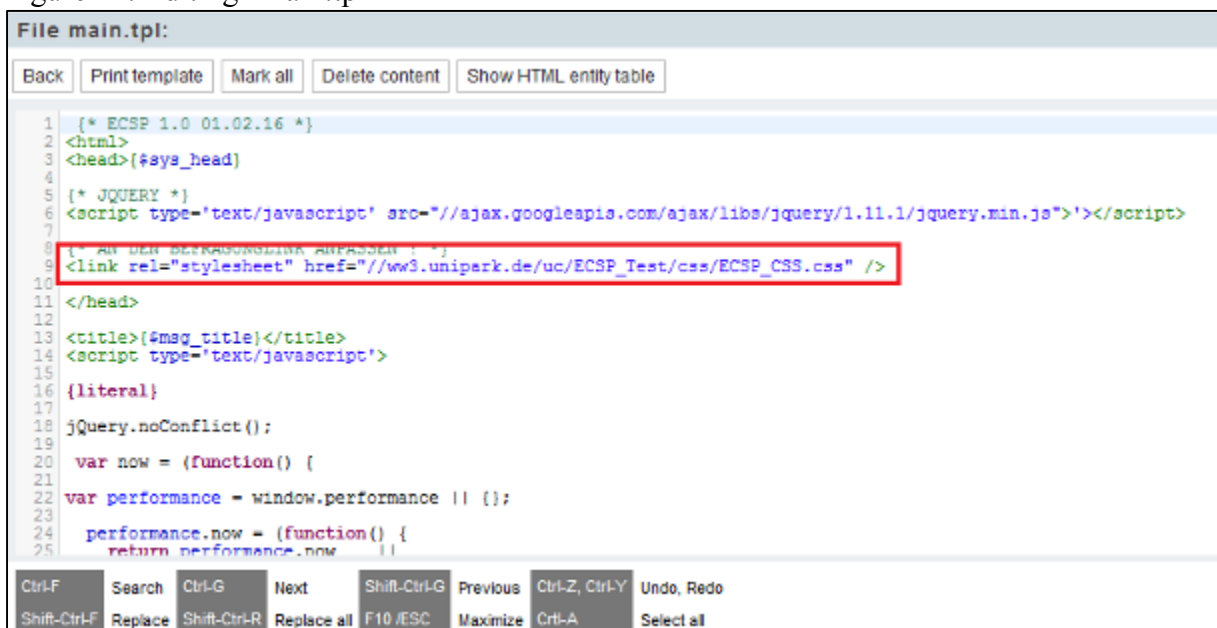
```
<link rel="stylesheet" href="//ww3.unipark.de/uc/ECSP_Test/css/ECSP_CSS.css" />
```

... is changed accordingly to ...

```
<link rel="stylesheet" href="//ww3.unipark.de/eee/questionnaire_Test/css/ECSP_CSS.css" />
```

... and is saved.

Figure 11: Editing “main.tpl”



Step 4 - Extending the survey pages to capture paradata

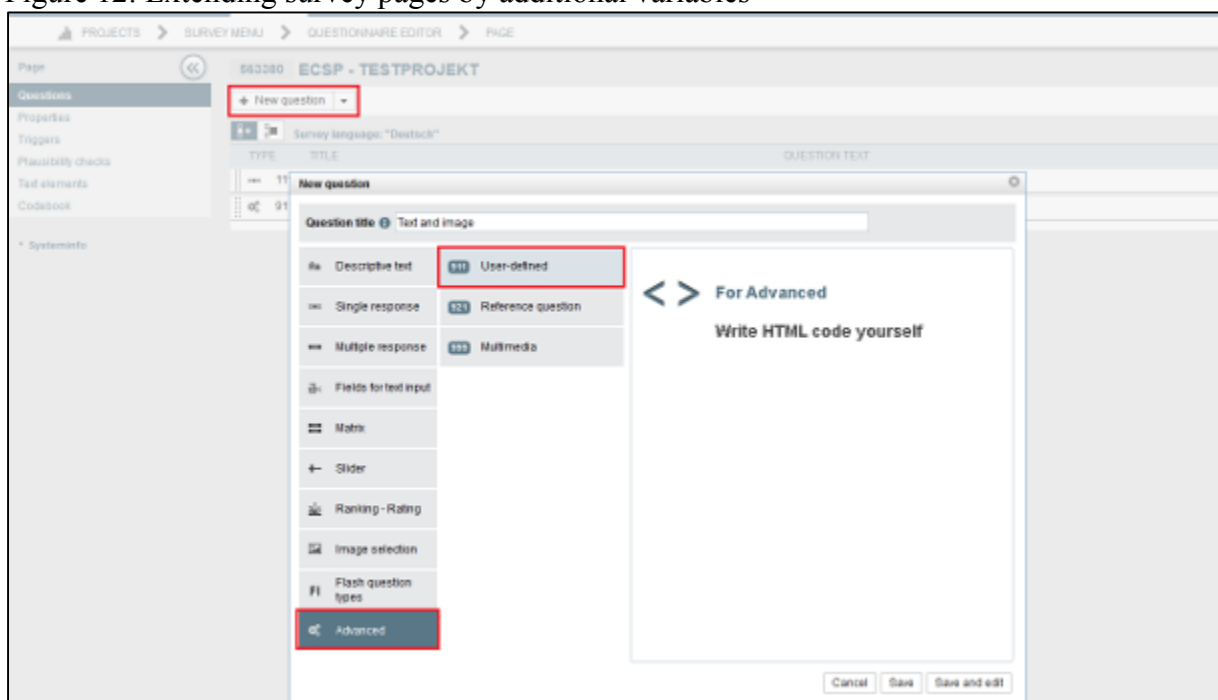
In this step the survey pages are extended by additional variables to be able to capture paradata. For this purpose, an additional and for the respondent's invisible question is put on every survey page (This question is built up identically for every survey page and adapts itself by simply copying it to another page).

[Survey Menu → Questionnaire editor] [Questionnaire page]

On any page, preferably the first page of the survey, an additional question of the type 911 is added. The title of the question is freely eligible (see Figure 12).

[New question → Advanced → User-defined 911]

Figure 12: Extending survey pages by additional variables



The following program code should be pasted in the area for HTML code (see Figure 13):

```
<input type=hidden name="varZeitDiff" value="v_XX1">
<input type="hidden" name="v_XX1" value="">

<input type=hidden name="varZeitStart" value="v_XX2">
<input type="hidden" name="v_XX2" value="">

<input type=hidden name="varTArrayOut" value="v_XX3">
<input type="hidden" name="v_XX3" value="">

<input type=hidden name="varYArrayOut" value="v_XX4">
<input type="hidden" name="v_XX4" value="">

<input type=hidden name="varXArrayOut" value="v_XX5">
<input type="hidden" name="v_XX5" value="">

<input type=hidden name="varClickTArrayOut" value="v_XX6">
<input type="hidden" name="v_XX6" value="">

<input type=hidden name="varClickYArrayOut" value="v_XX7">
```

```

<input type="hidden" name="v_XX7" value="">

<input type="hidden" name="varClickXArrayOut" value="v_XX8">
<input type="hidden" name="v_XX8" value="">

<input type="hidden" name="varSizeTArrayOut" value="v_XX9">
<input type="hidden" name="v_XX9" value="">

<input type="hidden" name="varSizeYArrayOut" value="v_X10">
<input type="hidden" name="v_X10" value="">

<input type="hidden" name="varSizeXArrayOut" value="v_X11">
<input type="hidden" name="v_X11" value="">

<input type="hidden" name="varScrollTArrayOut" value="v_X12">
<input type="hidden" name="v_X12" value="">

<input type="hidden" name="varScrollYArrayOut" value="v_X13">
<input type="hidden" name="v_X13" value="">

<input type="hidden" name="varZoomArrayOut" value="v_X14">
<input type="hidden" name="v_X14" value="">

<input type="hidden" name="varKeyTArrayOut" value="v_X15">
<input type="hidden" name="v_X15" value="">

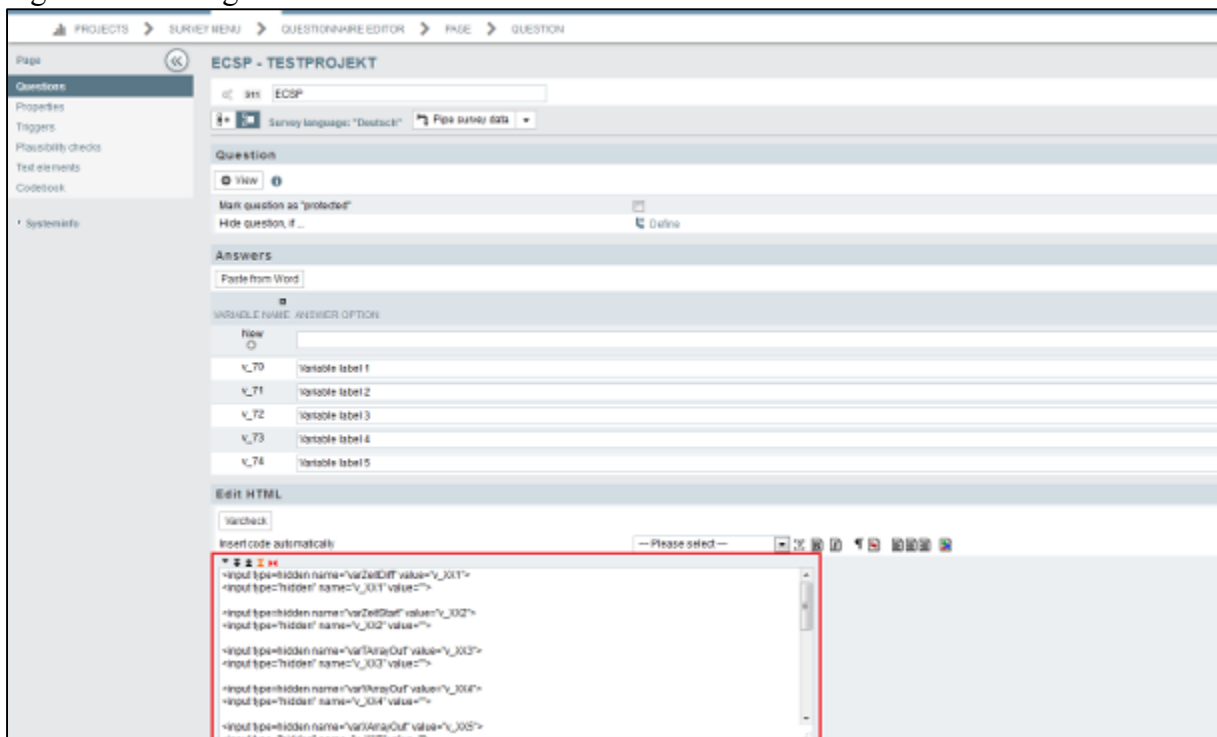
<input type="hidden" name="varKeyCodeArrayOut" value="v_X16">
<input type="hidden" name="v_X16" value="">

<input type="hidden" name="varFocusTArrayOut" value="v_X17">
<input type="hidden" name="v_X17" value="">

<input type="hidden" name="varFocusArrayOut" value="v_X18">
<input type="hidden" name="v_X18" value="">

```

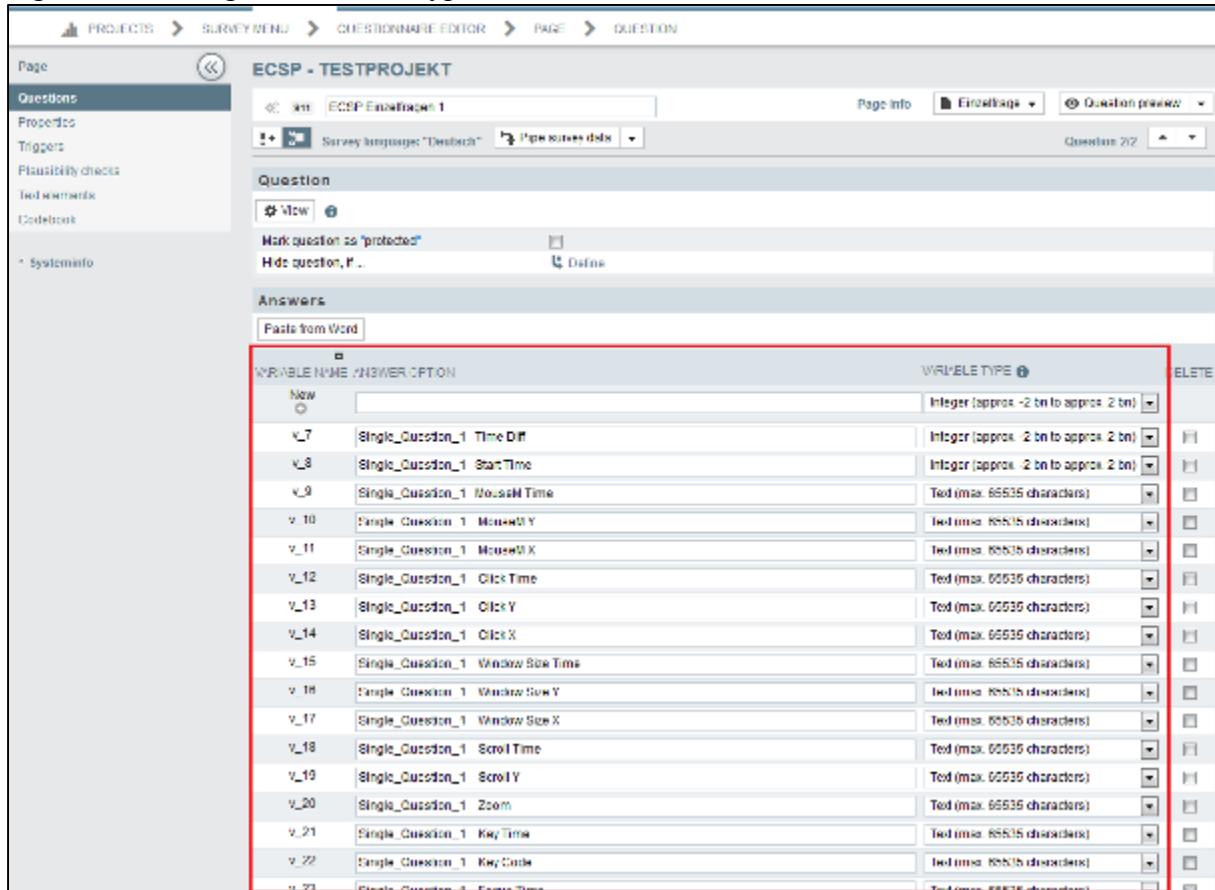
Figure 13: Editing the code of the additional variables



In the following, a total number of 18 variables should be created. In this variables the information of the captured paradata will be stored. For the first two variables, the “variable type” must be “Integer (approx. -2 bn to approx. 2 bn)”, all following variables should be able to collect text information “Text (max. 65535 characters)”.

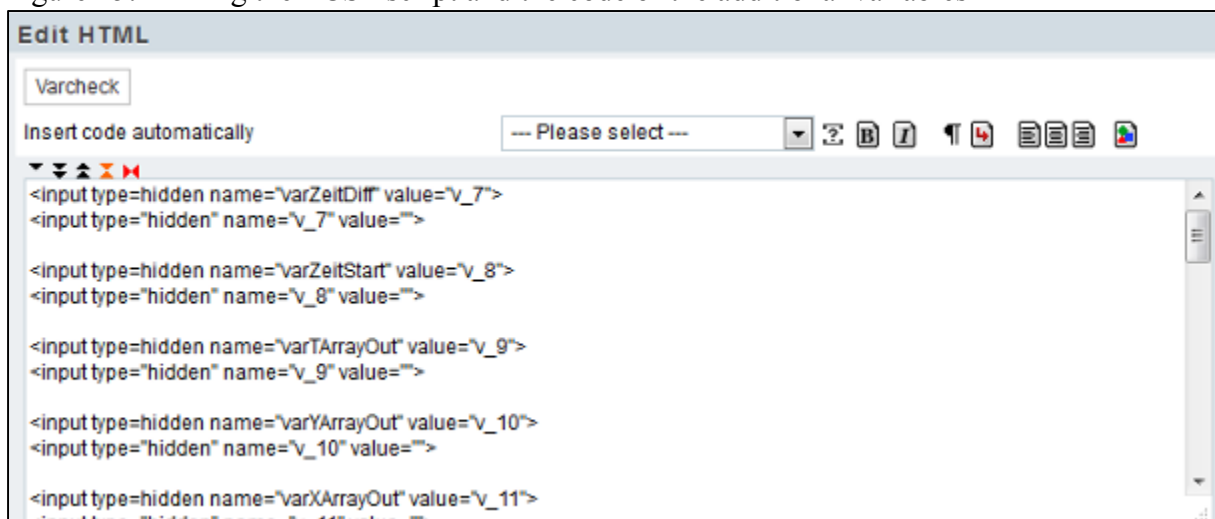
The descriptions of the single variables can be freely chosen. However, a description must be entered, because the variable cannot be created, otherwise. The individual variables are, however, bound to certain paradata information. For example, time measurements are captured by the first two variables (see Figure 14).

Figure 14: Editing the “variable type” of the additional variables



Afterwards a link between the ECSP script and the just provided variables is created. The place holders contained in the HTML script (“XX1” to “X18”; two place holders for each case) are substituted (in descending order of the variables) with their labels (in this example it concerns the variables “v_7” to “v_24”. See Figure 15).

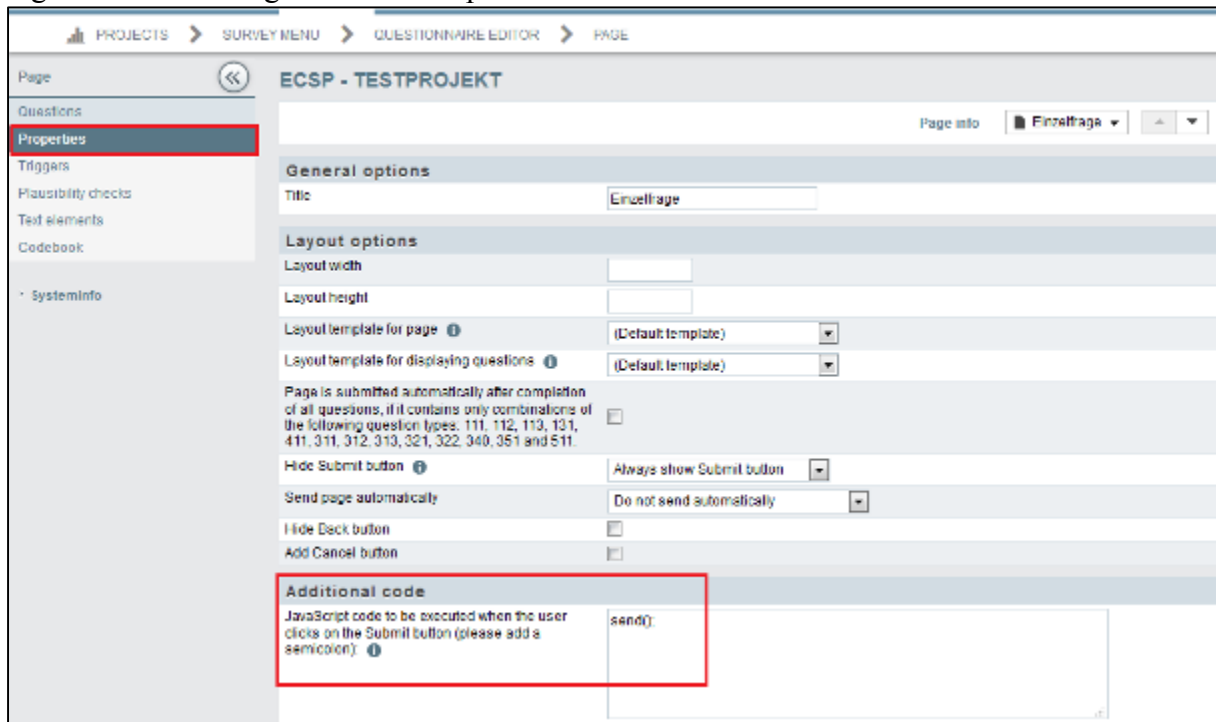
Figure 15: Linking the ECSP script and the code of the additional variables



Finally, it is essential to associate the paradata script with the “next button” of the survey page. Therefore, the command `send();` must be entered on each of the survey pages in the submenu “Properties” in the input field “Additional code”. The integration of the CSS-template can be checked in the “Header section” (see Figure 16).

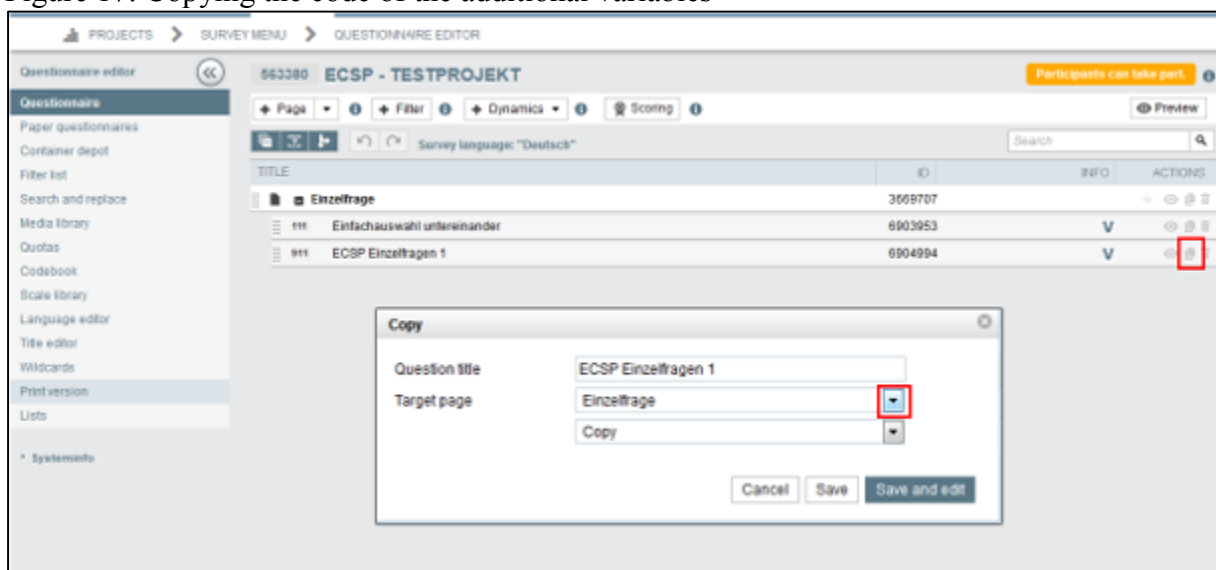
[Survey Menu → Questionnaire editor → Page → Properties]

Figure 16: Allocating the ECSP script and “next button”



Now this survey page is ready to capture paradata. The additional question can be pasted to all other survey pages (see Figure 17).

Figure 17: Copying the code of the additional variables



If necessary, the labels of the items can be adjusted. The allocation between the HTML script and the newly generated variables is automatically carried out by EFS-Survey/Unipark. In the submenu “page properties” the additional command `send()`; must be deposited on every single survey page separately!

Step 5 - ECSP function check

Whether the integration of ECSP has run successfully, can be checked after pretesting the survey. In addition, the content of the paradata variables can be checked in the submenu “Open-ended answers”. Both Integer values can be checked in the area “Online statistics” (see Figure 18 and 19).

[Survey Menu → Statistics → Open-ended answers]

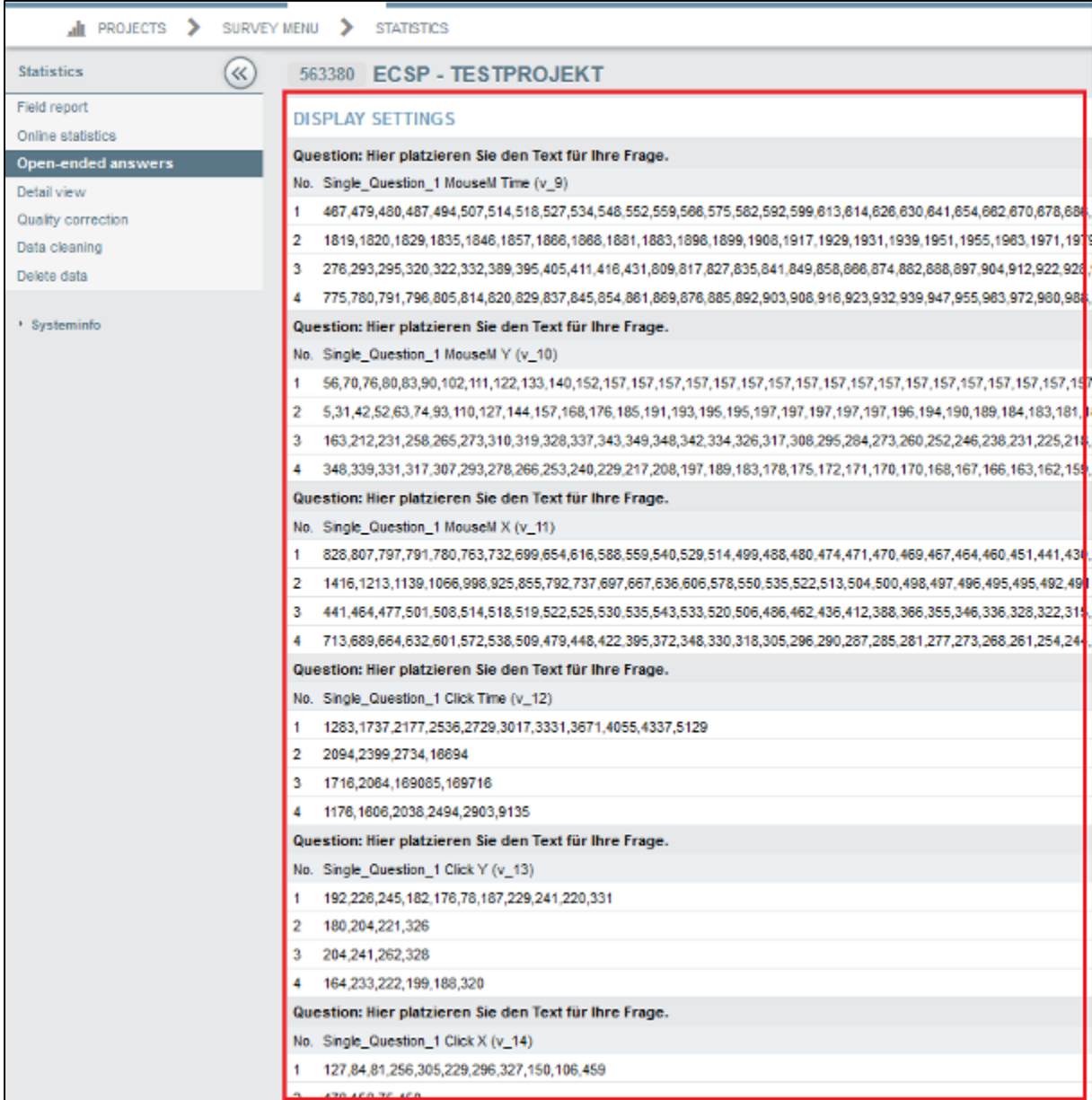
[Survey Menu → Statistics → Online statistics]

Figure 18: ECSP function check I

The screenshot shows the 'Open-ended answers' section of the EFS-Survey/Unipark interface. The breadcrumb navigation is 'PROJECTS > SURVEY MENU > STATISTICS'. The survey title is '063380 ECSP - TESTPROJEKT'. The selected language is 'Deutsch'. The section is titled '3 Pages: Einzelfrage'. The list of variables includes:

- ECSP Einzelfragen 1 -v_8: Single_Question_1 MouseM Time
- ECSP Einzelfragen 1 -v_10: Single_Question_1 MouseM Y
- ECSP Einzelfragen 1 -v_11: Single_Question_1 MouseM X
- ECSP Einzelfragen 1 -v_12: Single_Question_1 Click Time
- ECSP Einzelfragen 1 -v_13: Single_Question_1 Click Y
- ECSP Einzelfragen 1 -v_14: Single_Question_1 Click X
- ECSP Einzelfragen 1 -v_15: Single_Question_1 Window Size Time
- ECSP Einzelfragen 1 -v_16: Single_Question_1 Window Size Y
- ECSP Einzelfragen 1 -v_17: Single_Question_1 Window Size X
- ECSP Einzelfragen 1 -v_18: Single_Question_1 Scroll Time
- ECSP Einzelfragen 1 -v_19: Single_Question_1 Scroll Y
- ECSP Einzelfragen 1 -v_20: Single_Question_1 Zoom
- ECSP Einzelfragen 1 -v_21: Single_Question_1 Key Time
- ECSP Einzelfragen 1 -v_22: Single_Question_1 Key Code
- ECSP Einzelfragen 1 -v_23: Single_Question_1 Focus Time
- ECSP Einzelfragen 1 -v_24: Single_Question_1 Focus Code

Figure 19: ECSP function check II



Technical Details ECSP

The following EFS-Survey/Unipark question types can be used in conjunction with ECSP and are checked for functionality and correct visualization:

Descriptive text	Type 998	Text and image
Single response	Type 111	Single response list (vertical)
	Type 112	Single response list (horizontal)
	Type 113	Single response list (scale above)
Multi response	Type 121	Multiple response list (vertical)
	Type 122	Multiple response list (horizontal)

Fields of text input	Type 141	Text field (single row)
	Type 143	Multi text field (vertical)
	Type 144	Multi text field (horizontal)
Matrix	Type 311	Standard Matrix 1
	Type 312	Standard Matrix 2
	Type 313	Standard Matrix 3
	Type 321	Scale Matrix 1
	Type 322	Scale Matrix 2
	Type 340	Semantic differential
	Type 351	Double-Scale Matrix
	Type 362	Checkbox Matrix

The following list contains a short description of each paradata type:

1. Processing time: Time measurement in milliseconds (integer) of the time of the complete page setup up until sending the survey page by using the "next button".
2. Start time: Time of the complete page setup up (integer). Measured in Unix time (also known as POSIX time or Epoch time) number of milliseconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.
3. Timestamp mouse movements: Time measurement for each captured coordinate of the cursor in milliseconds. The values are stored separated by commas in a string variable.
4. Y-coordinate mouse movements: Measurement for each captured y-coordinate of the cursor in pixel (top left coordinate zero). The values are stored separated by commas in a string variable.
5. X-coordinate mouse movements: Measurement for each captured x-coordinate of the cursor in pixel (top left coordinate zero). The values are stored separated by commas in a string variable.
6. Timestamp mouse clicks: Time measurement for each captured coordinate of a mouse click in milliseconds. The values are stored separated by commas in a string variable.
7. Y-coordinate mouse movements: Measurement for each captured y-coordinate of a mouse click in pixel (top left coordinate zero). The values are stored separated by commas in a string variable.
8. X-coordinate mouse movements: Measurement for each captured x-coordinate of a mouse click in pixel (top left coordinate zero). The values are stored separated by commas in a string variable.
9. Timestamp screen size: Time measurement for each captured screen size or screen size change. The values are stored separated by commas in a string variable.
10. Y-coordinate width screen size: Measurement for each captured screen size or screen size change in pixel. The values are stored separated by commas in a string variable.

11. X-coordinate width screen size: Measurement for each captured screen size or screen size change in pixel. The values are stored separated by commas in a string variable.
12. Timestamp scrolling (vertical): Time measurement for each captured scrolling. The values are stored separated by commas in a string variable.
13. Scrolling (vertical): The value indicates the pixel which is displayed on the vertical zero line of the browser. The values are stored separated by commas in a string variable.
14. Zoom level. The value indicates a change in the browser zoom, the value 1 corresponds to a zoom level of 100% in Firefox.
15. Timestamp keystroke: Time measurement for each captured keystroke in milliseconds. The values are stored separated by commas in a string variable.
16. Code keystroke: Keystrokes are recorded in ASCII code. The values are stored separated by commas in a string variable.
17. Timestamp Survey Focus: Time measurement for leaving the web survey for a certain time period. The values are stored separated by commas in a string variable.
18. Survey Focus Code: Leaving the survey is measured by the value 0. Upon return, the value of 1 is given. The values are stored separated by commas in a string variable.

Not yet implemented features and limitations

Survey projects can be realized with ECSP only in the “classical layout” and not in the “adaptive layout” of EFS-Survey/Unipark. This is due to the use of a standardization of the browser settings. In addition, not all question types can be consistently visualized across different browsers so far (e.g. items with sliders and content based on Flash).

The transferability of the approach used by ESCP on mobile devices with touch screen could not yet be implemented and reviewed for all paradata types. Up to now, verified results exist only for the measurement of processing and transfer times for these devices.

So far, the data transfer occurs synchronically between clients and the EFS-Survey/Unipark server. An asynchronous communication (e.g. via AJAX) would be desirable, in order to reduce the transmission time between the survey pages or in order to be able to collect information about the respondent’s behavior right before closing an not submitted survey page.

Acknowledgements

I would like to thank Steffen M. Kühnel for his advice and support at all levels. I would also like to thank o Jürgen Leibold as well as Jan Karem Höhne for the inspiring discussions about the topic of paradata.

References

- Blank, G., and Rassmussen, K. (2004). The Data Documentation Initiative. The Value and Significance of a Worldwide Standard. *Social Science Computer Review*, 22(3):307-318.
- Callegaro, M. (2013). Paradata in Web Surveys. In *Improving Surveys with Paradata. Analytic Uses of Process Information*, ed. F. Kreuter, 261-279. Hoboken, NJ: John Wiley & Sons.
- Couper, M. P. (2000). Usability Evaluation of Computer-Assisted Survey Instruments. *Social Science Computer Review*, 18(4):384-396.
- Couper, M. P., Kreuter, F., and Lyberg, L. (2010). The Use of Paradata to Monitor and Manage Survey Data Collection. In *Proceedings of the Section of Survey Research Methods*, 282-296. Alexandria, VA: ASA, American Statistical Association – JSM 2010.
- Couper, M. P., and Lyberg, L. (2005). The Use of Paradata in Survey Research. *Proceedings of the 55th Session of the International Statistical Institute*, Sydney, Australia.
- Couper, M. P., and Singer, E. (2013). Informed Consent for Web Paradata Use. *Survey Research Methods* 7(1):57-67.
- DGS (2016). Ethik-Kodex der Deutschen Gesellschaft für Soziologie. Webpage: <http://www.soziologie.de/de/die-dgs/ethik/ethik-kodex.html> (last update: 12.02.2016).
- ESOMAR (2015). ESOMAR/GRBN Guideline on Online Sample Quality. Webpage: <https://www.esomar.org/uploads/public/knowledge-and-standards/codes-and-guidelines/ESOMAR-GRBN-Online-Research-Guideline-October-2015.pdf> (last update: 12.02.2016).
- Heerwegh, D. (2003a). Explaining Response Latencies and Changing Answers Using Client-Side Paradata from a Web Survey. *Social Science Computer Review* 21(3):360-373.
- Heerwegh, D. (2003b). The CSP Project. Webpage: <https://perswww.kuleuven.be/~u0034437/public/csp.htm#current> (last update: 12.02.2016).
- Heerwegh, D. (2011). Internet Survey Paradata. In *Social Behavioral Research and the Internet. Advances in Applied Methods and Research Strategies*, ed. M. Das, P. Ester, L. Kaczmirek, 325-348. New York, NY: Taylor and Francis.
- Kaczmirek, L. (2005). A Framework for the Collection of Universal Client Side Paradata (UCSP). Webpage: <http://kaczmirek.de/ucsp/ucsp.html> (last update: 12.02.2016).
- Kaczmirek, L. (2009). Human Survey-Interaction: Usability and Nonresponse in Online surveys. Mannheim: MADOC (Mannheim Electronic Document Server).
- Kreuter, F. (2013). Improving Surveys with Paradata: Introduction. In *Improving Surveys with Paradata. Analytic Uses of Process Information*, ed. F. Kreuter, 1-9. Hoboken, NJ: John Wiley & Sons.
- Mohler, P. P., Pennell, B. E., and Hubbard, F. (2008). Survey Documentation: Towards Professional Knowledge Management in Sample Surveys. In *International Handbook of*

Survey Methodology, ed. E. De Leeuw, J. Hox, D. Dillman, 403-420. New York, NY: Lawrence Erlbaum.

O'Reilly, J. (2009). *Paradata and Blaise: A Review of Recent Applications and Research*. Paper presented at the International Blaise Users Conference (IBUC), Latvia.

Resig, J., and Bibeault, B. (2013). *Secrets of the JavaScript Ninja*. New York, NY: Manning Publications Co.

Scheuren, F. (2005). *Paradata from Concept to Completion*. Proceedings of the Statistics Canada Symposium. Methodological Challenges for Future Information Needs.

Sundgren, B. (1973). *An Infological Approach to Data Bases* (Urval nr 7). Stockholm: K. L. Beckmans Tryckerier AB.