Fine-Grained Analysis of Web Tasks through Data Visualization

Gennaro Costagliola and Vittorio Fuccella

Department of Mathematics and Informatics,
University of Salerno
{gencos,vfuccella}@unisa.it

Abstract. This paper presents an approach for monitoring several important aspects related to user behaviour during the execution of Web tasks[1]. The approach includes the tracking of user interactions with the Web site and exploits visual data mining to highlight important information regarding Web application usage. In particular, our approach intends to be a *natural heir* of the approaches based on *clickstream* visualization, by integrating them with the visualization of page-level data and by improving them with the definition of ad-hoc *zoom* and *filter* operations. Furthermore, we present a theoretical framework to formally define our proposal. Lastly, in order to test the approach, a simple case-study for a particular practical usability evaluation has been carried out. To this aim, we built a prototypal system composed of a tracking tool, responsible for tracking user interactions and a visualization tool for task analysis.

1 Introduction

The analysis of user behaviour during Web navigation is a potentially fruitful source of important information. In particular, it has been extensively used for improving usability, but also to provide a better support to task users, i.e. by improving Web browsers and navigation tools [1], and to discover behavioural patterns in a given category of users [2].

The recent AJAX [3] technologies, provide us with detailed information regarding user behaviour, by allowing us to capture user interface events triggered on the client-side of Web applications. For example, with AJAX we can easily capture and record users' behaviour on complex interaction systems, such as Web forms, which are the primary medium for user input on the web [4].

Unfortunately, since interface events are not included in traditional server-side logs, they have scarcely been considered in the analysis of on line tasks for a long time, despite the richness of information regarding user behaviour they convey. From server logs, we can only elicit the *clickstream*, which is the

[1] This research has been partially supported by the grant "cofinanziamento per attrezzature scientifiche e di supporto, grandi e medie (2005)" from the University of Salerno.

M. Gaedke, M. Grossniklaus, and O. Díaz (Eds.): ICWE 2009, LNCS 5648, pp. 1–15, 2009.

recording of what a computer user clicks on while Web browsing. Past systems, such as *WebQuilt* [5], in fact, relied on *clickstream* visualization, to graphically show Web users behaviour on Web tasks.

The availability of information on the user interactions with the Web interfaces should be effectively used for the previously mentioned purposes, in particular for usability evaluation. In literature, some attempts to integrate web server logs with client-side data for analysis purposes are present [6] [7], but, to our knowledge, no complete and effective visualization methods for Web task analysis, based on client-side data, have been proposed. In particular, usability analysis still largely employs the *think out loud* protocol, which has several disadvantages: to some extent, a user has to interrupt operations, when commenting; evaluators are expected to have special knowledge; subjects do not necessarily comment [8].

In this paper we present an approach for monitoring several important aspects related to user behaviour during the execution of Web tasks. The approach includes the tracking of user interactions with the Web site and exploits data visualization to highlight important information regarding Web application usage. In particular, our approach intends to be a *natural heir* of the approaches based on *clickstream* visualization: while for the existing approaches *clickstream* visualization is the final result, in our system its graph chart is only the starting point of a more comprehensive visual data mining process. To elaborate, it is our approach's *first level* of analysis. By applying a *zoom in* operation, we can deepen the analysis to a *second level* chart, showing the page-level interactions of the users. A *filter* operation is also defined. It allows the analyzer to visualize only a selection of the currently visualized user sessions involved in the task.

In order to define in an unambigous way the charts and the operations of filtering and zooming, the paper presents a theoretical framework. These operations, largely employed in visual data mining [9] are defined in the context of our approach. Lastly, we give a first glance at the way our approach can be used for detecting usability problems in Web pages. To this aim, we built a prototypal system and used it in a real life example. The example shows how our approach, by performing simple visual interactions, aids the analyst in detecting usability problems in Web tasks.

The rest of the paper is organized as follows: Section 2 contains a brief survey on works related to ours. The whole approach is presented in Section 3. The theoretical framework is presented in Section 4, whereas, in Section 5, we discuss the results of a case-study application. Several final remarks and a brief discussion on future work conclude the paper.

2 Related Work

Several approaches can be found in the literature that describe how to capture and visualize user behaviour in Web tasks. Most of them only rely on server log analysis. In recent years, there have been several attempts of integrating web server logs with client-side data, but only a few of them have resulted in an effective visual data mining method for user behaviour analysis.

Among the earliest methods there are those exploiting *clickstream* visualization. *WebQuilt* (2001), [5] employs *clickstream* visualization for highlighting the most used paths by the users to accomplish a given web task. A variant of the above method, called *DiskTree*, transforms the *clickstream* graph in a 2D tree, generally by simply running the Breadth First Search algorithm on it, and visualizes it in a disk, such that the root is located in the centre and the last level nodes on the circumference of the disk. Chi [10] used this approach for discovering usability defects in Web sites. Chen et al. [11] have improved this method by introducing operations and by describing a web graph algebra to manipulate and combine web graphs.

Specific approaches for the analysis of e-commerce systems capturing and visualizing user interface interactions are presented in [12,13,14]. In [12] it is presented a case study based on *ClickViz*. This system integrates *clickstream* data with user demographic information. The other two ones refine *clickstream* data by selecting specific web merchandising events from the web server log, such as: product impression, clickthrough, basket placement and purchase. In [14] the analysis is oriented to the product evaluation, which is shown through a starfield display [15], where it is possible to verify that user interest for the product is in tune with its visibility in the site. In [13], instead, an interactive parallel coordinate system is used to interpret and explore *clickstream* data, checking, i.e., how many sessions lead to a purchase and how many abandoned the site before.

Other approaches [16] [6] [8], focus on the use of mouse and gaze movements as implicit interest indicators. In particular, the paths of the mouse and of the user gaze are visualized in a video [8] or in a page screenshot [6] as lines overimposed to the image visualized on the screen at that moment.In [8] the approach is used in a usability study with 5 tasks where it is proved that the use of a video improves the think out loud protocol that would miss 11.6% of the entire user comments.

From a visualization point of view, our approach proposes a system similar to those visualizing the *clickstream*, significantly improving previous approaches with the appropriate integration of elements that visualize the additional client-side tracking information. This study originates from our previous experience on studying learners' strategies to complete tests by tracking their interactions with Web-based learning systems [17]. The current work is then a generalization of the previous one from a specific domain (e-learning) to the general case of web task analysis.

3 The Approach

In this section, we describe the approach for the analysis of user behaviour during online tasks. In particular, we have devised a new symbolic data visualization strategy, which is used within a KDD process to graphically highlight behavioral patterns related to the users activity in online tasks. The main two steps of our approach are *data collection* and *data visualization*.

3.1 Data Collection

The proposed approach aims to gather data concerning the users' activities during Web tasks. Gathering occurs by simply recording the interactions of the users with the Web site interface during navigation.

The data collection can be realized in a laboratory, with selected users, or during Web site usage, with real users. The case-study application results shown in section 5, refer to the first method only. The collected data carries information about the user session, the sequence of the pages loaded in the browser (*clickstream*) and the interactions of the users with the interactive elements contained in the pages, such as links and form fields. In particular, the information about a specific web task accomplished by several users is recorded in a log. During a session, the operations realized by a user in a specific task produce the following data:

1. information about the browser session: sessionID, IP address, username, information about the user agent, sequence of visited pages;
2. information about the visit of single pages: url of the page and timestamp of the loading and of the unloading events; the mouse movements and clicks, scrolling; presence and duration of inactivity time intervals (no interactions) during the page visit. Furthermore, the whole HTML code of the visited pages and the referenced resources are recorded;
3. events generated by the user interaction with the widgets of a page: duration of the interaction and sequence of events occurring in that interaction;

Given the hierarchical nature of the gathered data (session, page, page elements), we store data in XML format instead of storing it in plain text as in the usual log files. In this way the input for the next step (data visualization) is already partially elaborated.

3.2 Data Visualization

Data visualization is an important tool for Web task analysis. It has been widely employed for *clickstream* analysis, so far. Nevertheless, *clickstream* only includes high level information: it does not allow a deep analysis of user behaviour. In our approach, the data gathered in step 1 are visualized through a suite of charts showing the user actions during the web tasks at various levels of detail. The charts are analyzed both top-down and bottom-up following the different levels. The analysis from the higher level starts visualizing cumulative and most significant data of the whole task, and proceeds to more detailed views through *zoom* and *filter* operations applied by the analyzer through simple interactions with the charts. The *filter* operation can also be applied to the detailed views to expose particular user behaviours and, if needed, the analyzer can return to higher level views to consider the context in which these behaviours occur.

Our approach includes the visualization of charts, showing cumulative information of the sessions of the users participating to a given task, at two levels of deepening.

The **first level chart** represents *Clickstream* through a direct graph. The node set of the graph includes all the pages visited by at least one user while each

edge is a transition between two pages. In our approach, as shown in Figure 6(i), to better associate the graph nodes to the visited pages, each node is represented through a thumbnail of the associated page. The color of the nodes background is forced to be in a gray scale (in order to represent the average time spent on the page) even in the event of a different color indicated by the Web page style settings. A direct edge is present between two nodes if at least one user has visited the pages represented by the two nodes in succession.

In our approach, pages requested from the same URL can be associated to different graph nodes. This happens since dinamically produced Web pages are often different, even when obtained by requesting the same URL. Two pages are regarded as different versions of the same page if their internal link structure is different. Graph nodes are visually arranged in a matrix. Nodes corresponding to different versions of the same page are shown in the same column, ordered descendingly by the number of sessions passing through the node. Further optimizations are performed in order to improve chart readability.

The **second level chart** is a deepening of the previous. It shows information regarding a visit of a single page. The used visual metaphore is a graph, as before. Nevertheless, this time the nodes represent the widgets (generally links and form fields) at least one user has interacted with. The chart is shown in transparency on the screenshot of the page, in order to highlight the widgets. A direct edge is present between two nodes if at least one user has interacted with the widgets represented by the two nodes in succession, that is, the user focus has directly passed from the first to the second widget, with no inner focuses on a third widget. At this level, previous and next pages are reported in the chart in two different columns, on the left and on the right of the currently zoomed page.

The association between visual cues and usage metrics is the following:

- **Node border thickness** represents the number of sessions passing through the node. A thicker border is for a node with a greater number of user accesses.
- **Node border color** represents the average number of passages through the node for each session (calculated only on the sessions which have at least one passage through the node). The color is reported in red scale: a darker red is for more passages.
- **Node internal background color** represents the average time spent on the node (in case of multiple passages in the same sessions, their durations are summed). The color is reported in gray scale: a darker gray is for a longer time.
- **Edge thickness** represents the number of sessions performing the transition.
- **Edge color** represents the average number of passages through the edge for each session (calculated only on the sessions which have at least one passage through the edge). The color is reported in gray scale: a darker gray (tending to black) is for more passages.

Since, compared to previous approaches, we increased the number of visual cues, we also made an effort to use them "consistently", thus guaranteeing a lower

cognitive load while interpreting the charts [18]. Firstly, the above association is valid in both levels. Furthermore, other solutions have been adopted: cues on nodes and edges are always associated to measures related to pages and transitions, respectively; line thickness, line color and internal color are always associated to number of sessions, average number of passages in a session and time of a visit measures, respectively.

It is possible to go back and forth through the levels by applying *zoom* (*in* and *out*) operations. Furthermore, *filter* operations can be applied in order to construct the graph only by using a selection of the currently visualized user sessions involved in the task. In particular, the subset of the sessions which satisfy a selection condition: nodes and edges can be marked with *inclusive* or *exclusive* ticks, indicating that the sessions underlying the selected nodes and edges are included or excluded, respectively. *Inclusive* and *exclusive* ticks can be interactively added on the graph. The *inclusive filter* is used to select only the sessions passing through the marked elements, whilst the *exclusive filter* is used to select all the sessions except those passing through the marked elements. An *inclusive filter* can be applyed, i.e. to an outgoing edge on a node representing an error page in first level chart, in order to understand how a set of users recovered from the error. Conversely, an *exclusive filter* can be applied to a node representing the submit button of a form in the second level chart, in order to visualize only the form filling patterns of the users who abandoned the form (users who never submit the form). A sample of application of such a *filter* is shown in Figure 1.

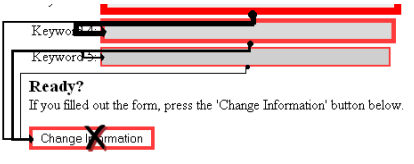

Fig. 1. Sample of application of a *filter*. The submit button of the form has been marked with an *exclusive* tick.

4 Theoretical Framework

In this section we provide a framework to formally define the operations of *filter* and *zoom in*. In order to do so we need to explicitely define the notions of session and page-session and, then, model the first and second level charts as labelled graphs.

4.1 The First Level Chart

Given a set of pages A, a *session* S on A is defined by a sequence of pages $(p_0, p_1, \ldots, p_n)$ in A, where p_0 and p_n are the entry and exit pages, respectively, and the set of all the transitions (p_i, p_{i+1}) of successive pages in the sequence.

A *first level chart* for a set of sessions Σ is defined as a labelled direct graph $G_\Sigma(V, E)$ where V is the union set of all the pages occurring in the sessions, i.e., $V = \cup p$, for each $p \in S$ and for each $S \in \Sigma$, and E is the union set of all the transitions occurring in the sessions, i.e., $E = \cup(p_i, p_{i+1})$, for each $(p_i, p_{i+1}) \in S$ and for each $S \in \Sigma$.

Each transition and page in the graph is labelled by a tuple. These tuples generally represent the metrics to keep track of, such as the ones introduced in subsection 3.2. In this section, for sake of simplicity we will only refer to the label w of a transition t representing the number of sessions containing at least one occurrence of the transition, i.e., $w_t = |\{S \in \Sigma | t = (p_i, p_{i+1}) \in S\}|$.

For example, let us consider the set of pages $A = \{p, q, r, s, u\}$ and the following set of sessions on A:

$\{S_1 = (p, q, s), S_2 = (p, r, p, q, s), S_3 = (p, r, p, q, s), S_4 = (p, u, p, q, s)\}$

The resulting labelled graph is shown in Figure 2.

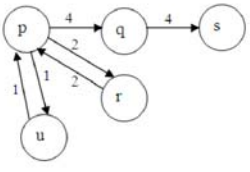

Fig. 2. A labeled direct graph modeling a first level chart

In the following we will use the term 'element' when referring to either a page (node) or a transition (edge) in a labeled direct graph.

The session filter operations. To define *filters* on a set of sessions we start by giving the notion of a *filter* based on a single element of the graph. We will then use this primitive *filter* to provide a general definition of complex *filters*.

Given a set of sessions Σ and an element e in G_Σ, we denote with Σ_e the set of all the sessions in Σ including e.

If we consider the previous example, Σ_r is $\{S_2 = (p, r, p, q, s), S_3 = (p, r, p, q, s)\}$ and $\Sigma_{(u,p)}$ is $\{S_4 = (p, u, p, q, s)\}$, leading, respectively, to Figures 3(i) and 3(ii).

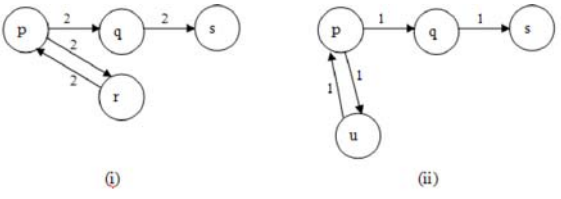

Fig. 3. Labeled direct graphs modeling Σ_r, (i), and $\Sigma_{(u,p)}$, (ii)

We can then define the *filter* operations: given a set of sessions Σ, a *filter on* Σ is defined by a set-theoretic expression on the set $\{\Sigma_e \in 2^{\Sigma} | \ e$ is an element in $G_{\Sigma}\}$ using the operations of union, intersection and complementation.

In particular, given a set M of (marked) elements in G_{Σ}, we can define, among the others, the following four *filters*:

- *union-inclusive filter*: for the selection of all the sessions of Σ with at least an element in M, (defined as $uif(\Sigma, M) = \cup_{e \in M} \Sigma_e$);
- *union-exclusive filter*: for the selection of all the sessions of Σ except for those with at least a marked element in M, (defined as $uef(\Sigma, M) = \Sigma - \cup_{e \in M} \Sigma_e$);
- *intersection-inclusive filter*: for the selection of all the sessions of Σ containing all the marked elements in M, (defined as $iif(\Sigma, M) = \cap_{e \in M} \Sigma_e$);
- *intersection-exclusive filter*: for the selection of all the sessions of Σ except for those containing all the marked elements in M, (defined as $ief(\Sigma, M) = \Sigma - \cap_{e \in M} \Sigma_e$).

Following the previous example, if we take $M = \{r, (u, p)\}$, as shown in Figure 4(i)), the application of the *union-inclusive filter* will filter out session S_1 producing the graph in Figure 4(ii), while the *union-exclusive filter* will filter out sessions S_2, S_3 and S_4 producing the graph in Figure 4(iii). The *intersection-inclusive filter* will filter out all the sessions while the *intersection-exclusive filter* will produce no filtering at all.

Note that the the *inclusive* and *exclusive filters* introduced in subsection 3.2 are formally defined by the intersection-inclusive and intersection-exclusive definitions above but the approach can be extended to support any other *filter*.

The session zoom-in operation. In order to define the *zoom-in* operation we first need to define the notion of page-session, i.e., the interactions of a user with a particular page extracted from the *clickstream*. This is very similar to the notion of session with the difference that we now deal with sequences of page-widgets.

Given a page p_i of a session S, a *page-session* P_{p_i} is a sequence $(w_0 = p_{i-1}, w_1, w_2, \ldots, w_{n-1}, w_n = p_{i+1})$, where each element w_i is a widget internal to page p_i. Pages p_{i-1} and p_{i+1} are the entry and the exit points of the

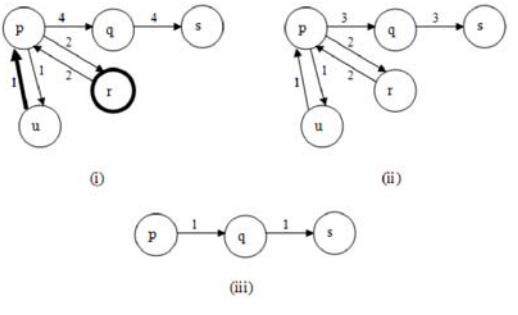

Fig. 4. Filter applications with $M = \{r, (u, p)\}$ (i): union-inclusion (ii) and union-exclusion (iii)

page-sessions, respectively. In the cases where p_{i-1} and p_{i+1} do not exist they are substituted by a dummy entry point and a dummy exit point, respectively. As with the definition of session, a page-session also includes all the transtions (w_i, w_{i+1}).

We can now give the definition of *zoom-in* operation: given a set of sessions Σ, and a page p of G_Σ, the *zoom-in operation* $zi(\Sigma, p)$ returns the the set of page-sessions $\Pi = \{P_p | p \in S \text{ and } S \in \Sigma\}$.

In other words a *zoom-in* operation extracts a page from the *clickstream* and makes explicit the user interactions with the page internal elements.

The following definitions are given for sake of completeness but they can be easily derived from the corresponding ones above.

4.2 The Second Level Chart

A *second level chart* for a set Π of page-sessions in p is formally described by a labeled direct graph $G_\Pi(V, E)$ where V is the union set of all the elements occurring in the page-sessions, i.e., $V = \cup_{w \in P} w$ for each $P \in \Pi$. E is the union set of all the transitions occurring in the page-sessions, i.e., $E = \cup(w_i, w_{i+1})$, for each w_i, $w_{i+1} \in S$ and for each $P \in \Pi$.

Each transition t in the graph G is labelled by a weigth w representing the number of page-sessions of Π containing at least one occurrence of the transition, i.e., $w_t = |\{P \in \Pi | t = (w_i, w_{i+1}) \text{ and } (w_i, w_{i+1}) \in P\}|$.

The page-session filter operations. Given a set of page-sessions Π and an element e of G_Π,we denote with Π_e the set of all the page-sessions in Π including e. The *filter* operations on Π can be analougously defined as done above on Σ.

5 The Case-study Application

In order to demonstrate the effectiveness of the approach, we have evaluated the usability of a simple Web task with respect to six students of the third year of the Laurea degree in Computer Science at the University of Salerno. The goal of this case-study is to gather preliminary data for a simple task and use our protoype to visually analyze them in order to discover eventual usability faults with respect to our sample. We chose a simple web task where few paths of few pages could be followed. Furthermore, the good variety of paths to be followed allowed us to reason on the cases of success, wrong input, and incomplete input.

Our users have been asked to perform an information seeking task on the Web site of the Italian ministry of universities and research (MIUR). In Italy, academic staff is hired by *concorsi* (national competitions). A national, discipline-related committee is responsible for choosing the winner candidate. The names of the committee's members are published through the Web site of the ministry. The starting URL for the procedure to search them is

http://reclutamento.miur.it/seleziona/commissioni.html.

A typical search can be performed by providing information such as the year of publication,the position and the discipline of the *concorso* and information

Fig. 5. The first form of the concorso committee search interface

about the organizing institution (university and/or faculty). All of the concorsi published in an year are grouped by sessions; in a year, from two to four sessions can be included. The search interface, shown in Figure 5, allows the seeker to specify one or more of the above data through two forms including five HTML select fields: *sessione*, *ruolo* (position) *ateneo* (university), *facoltà* and *settore disciplinare* (discipline). The two forms have identical structure but refer to the *concorsi* after and before the reform of the disciplines, respectively.

The statement of the task was the following: "search on the MIUR Web site the members of the committee of the *concorso* for the position of Researcher, discipline MAT/09 - Ricerca Operativa (Operative Research), published in 2007 at the University of Salerno, Faculty of Science".

The System. In order to carry out the case-study, we built a prototypal system. The system is composed of two main components: a *tracking tool*, responsible for tracking user interactions and saving the recording in a log file, and a visualization tool, responsible for aiding the analyzer to graphically inspect the log.

The tracking tool relies on the AJAX [3] technology in order to capture all of the users interactions with the user interface (running in the Web browser). It can be used as a framework to be instantiated in Web applications in order to track system users. Alternatively, it can be used as an external tool for tracking user behaviour on any Web site. The only requisite in the latter case, is to opportunely configure the Web browser to overcome security limitations, which do not allow the scripts to establish connections with external domains (cross-domain scripting security). The visualization tool is an *Integrated Analysis Environment*, in which a suite of interactive charts is combined with textual information presented under the form of tables, to perform a cross-data analysis.

Tracking. The test has been carried out in a laboratory equipped with PCs running a version of Firefox 3.x Web browser, opportunely configured to allow the interaction tracking. With an "authorization" by the webmaster of the analyzed site to insert our scripts in the pages, this step would have not been necessary. Relying on our approach, it was not necessary that the operator controlled the users during the execution of the task. Thus, the users were completely autonomous and performed the task at the same time. The operator was available

for answering possible users' questions. The task only lasted a few minutes (except the time necessary to configure the browser), and produced a log file sized less than 100Mb.

Analysis. The produced log has been analyzed with the *visualization tool*. The analysis begins with the visualization of the first level chart (see Figure 6(i)), which summarizes in a single view all of the sessions of the six users who performed the task. The users visited pages from the following three URLs (the prefix http://reclutamento.miur.it/seleziona/ has been cut for brevity):

- A: *commissioni.html*: static Web page containing the search form.
- B: *find_commiss.pl*: dynamic Web page containing the search results. Three different versions of this page have been visited by the users.
  - B1: page containing the expected search result, with a link to the committee's page;
  - B2: page with no results;
  - B3: page containing an error message (mandatory data not specified) and a back link to A.
- C: *commiss_names.pl*: dynamic Web page containing task's final results, that is, the names of the committee's members.

By observing the image, we immediately note that the internal color of node A is darker than other nodes. This suggests that the average time spent on the page with the form is significantly greater than the time spent in checking the search results. Furthermore, nodes B2 and B3 have a lighter color (less time spent) and a thinner border (less visits). The image summarizes the sessions of our six users. In the following, we report in bold face their usernames (as chosen by them) and the sequence of visited pages.

alessia A → B1 → C
gatto A → B2 → A → B1 → C
lella A → B1 → C
lulu A → B2 → A → B1 → C
mirco A → B3 → A → B1 → C
zanzibar A → B3 → A → B1 → C

The three nodes on top of the chart shown in Figure 6(i) are those belonging to the *main flow*, that is, the users must visit the A → B1 → C path in order to successfully accomplish the task. In the case of our test, all of the users did. This can be argued by both inspecting the above sequences and observing the chart. In particular, we note that the nodes corresponding to the pages A, B1 and C have the same border thickness. So do the edges connecting A with B1 and B1 with C. The presence in the chart of pages B1 and B2 is sign of an expected behaviour from the users. The presence of page B3, which contains an error message stating that the required *sessione* field has not been set, is, instead, sign of a usability problem: the *sessione* field is not adequately marked as required in page A. The only cue that differentiates it from the other fields is a "(please) select!" message in the field. Not enough, we can argue, considering that two users out of six left the field blank. Furthermore, the instructions for

Fig. 6. A selection of screenshots from the visualization tool interface. (i) The first level chart; (ii) the second level chart obtained by zooming-in on page A; (iii) the same chart after the application of an *inclusive filter* on the edge going from B3 to the zoomed page.

filling the form in the upper part of the page state that it is required to "select at least one field", without specifying which one of them is required.

To have a more detailed insight into the situation, hopefully discovering other usability issues with respect to our sample, we start performing a *zoom* operation on page A, obtaining the chart shown in Figure 6(ii). The chart visualizes the 10 page-sessions reported in the following. Here, each field's identifier is composed of a form identifier (0 or 1) and the field name, separated with a dot. The *START* element denotes the dummy entry point (see Section 4.1).

alessia START → 0.sessione → 0.ruolo → 0.settore → 0.sessione → 0.facolta → 0.ateneo → 0.cerca → B1
gatto START → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.sessione → 0.cerca → B2
gatto B2 → 0.sessione → 0.cerca → B1
lella START → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.facolta → 0.settore → 0.cerca → B1
lulu START → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.cerca → B2
lulu B2 → 0.sessione → 0.cerca → B1
mirco B3 → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.sessione → 0.cerca → B1
mirco START → 0.sessione → 1.sessione → 0.settore → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.cerca → B3
zanzibar START → 0.sessione → 0.ruolo → 0.facolta → 0.ateneo → 0.facolta → 0.settore → 0.cerca → B3
zanzibar B3 → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.cerca → B1

The page-sessions visualized in the image are all those passing through page A. Some sessions have more page-sessions satisfying this requirement, in particular all the sessions passing through B2 and B3, since the users had to fill the form again. To keep on analyzing the task, let's apply some *filters*. In particular, let's see what happens by applying an inclusive *filter* on the edge (B3, 0.sessione). By performing this action, we are only selecting the page-sessions that, coming from the error page (B3), have brought again to the starting page (A, the one with the form): A. The page-sessions are the following:

mirco B3 → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.sessione → 0.cerca → B1
zanzibar B3 → 0.sessione → 0.ruolo → 0.ateneo → 0.facolta → 0.settore → 0.cerca → B1

In the resulting chart shown in Figure 6(iii), all of the form fields are highlited (in red). This means that, coming from the error page, the form has been completely reset, forcing the user to re-enter all the input fields. In a well designed interface, the values of the fields should have been kept. This is the second usability problem detected with respect to our sample. By analyzing the code of page B3, we realize that the link back to page A has been coded by the following HTML element:

 Torna indietro

In order to keep the form values, the back link should have been implemented through a Javascript history.back() statement.

6 Conclusions

We have presented an approach for the analysis of Web tasks by means of Information Visualization. The approach significantly improves previous ones based on *clickstream* visualization. It enables several kinds of analysis in order to trigger the analyzer's attention on behavioral patterns of the users. In this way, the analyzer is provided with a powerful tool that lets him/her review the whole task. A theoretical framework has been defined in order to clearly define the basic elements of the approach and the semantics of the classical visual data mining operators such as *filters* and *zoom* with respect to our case. We have also presented a case-study to show how our approach aids the analyst in detecting usability problems in Web tasks.

We believe that much more patterns can be discovered than those highlited in this paper and that the approach can be used with more general objectives other than evaluating Web usability. Thus, we are planning to perform further experiments, aimed at discovering and classifying new Web usage patterns and at testing our approach in different case-studies. In particular, we are planning to test our approach with a larger number of users, trying also to address problems related to the visualization of large and dense graphs.

Furthermore, we would like to enrich the model with new visual representations to express further metrics not represented yet by the visual cues used in the current charts. Finally, we are adding new features to the prototype and testing it in order to obtain a stable and robust system.

References

1. Kellar, M.: An examination of user behaviour during web information tasks. PhD thesis, Dalhousie University, Dalhousie Univ., Halifax, Canada (2007)
2. Sellen, A.J., Murphy, R., Shaw, K.L.: How knowledge workers use the web. In: CHI 2002: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 227–234. ACM, New York (2002)
3. Murray, G.: Asynchronous javascript technology and xml (ajax) with the java platform (October 2006), `http://java.sun.com/developer/technicalArticles/J2EE/AJAX/`
4. Thompson, S., Torabi, T.: A process improvement approach to improve web form design and usability. In: DEXA 2007: Proceedings of the 18th International Conference on Database and Expert Systems Applications, Washington, DC, USA, pp. 570–574. IEEE Computer Society, Los Alamitos (2007)
5. Hong, J.I., Heer, J., Waterson, S., Landay, J.A.: Webquilt: A proxy-based approach to remote web usability testing. ACM Transactions on Information Systems 19, 263–285 (2001)

6. Atterer, R., Wnuk, M., Schmidt, A.: Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In: WWW 2006: Proceedings of the 15th international conference on World Wide Web, pp. 203–212. ACM, New York (2006)
7. Paganelli, L., Paternò, F.: Intelligent analysis of user interactions with web applications. In: IUI 2002: Proceedings of the 7th international conference on Intelligent user interfaces, pp. 111–118. ACM, New York (2002)
8. Nakamichi, N., Sakai, M., Shima, K., Hu, J., ichi Matsumoto, K.: Webtracer: A new web usability evaluation environment using gazing point information. Electron. Commer. Rec. Appl. 6(1), 63–73 (2007)
9. Keim, D.A.: Information visualization and visual data mining. IEEE Transactions on Visualization and Computer Graphics 8(1), 1–8 (2002)
10. Chi, E.H.: Improving web usability through visualization. IEEE Internet Computing 6(2), 64–71 (2002)
11. Chen, J., Sun, L., Zaïane, O.R., Goebel, R.: Visualizing and discovering web navigational patterns. In: WebDB 2004: Proceedings of the 7th International Workshop on the Web and Databases, pp. 13–18. ACM, New York (2004)
12. Brainard, J., Becker, B.: Case study: E-commerce clickstream visualization. In: IEEE Proceedings of Information Visualization 2001 (INFOVIS 2001), pp. 153–156. IEEE Computer Society, Los Alamitos (2001)
13. Cofino, T., Gomory, S., Lee, J., Podlaseck, M.: Method for graphically representing clickstream data of a shopping session on a network with a parallel coordinate system. U.S. Patent Office (2007), Patent number 7266510
14. Lee, J., Podlaseck, M.: Using a starfield visualization for analyzing product performance of online stores. In: EC 2000: Proceedings of the 2nd ACM conference on Electronic commerce, pp. 168–175. ACM, New York (2000)
15. Ahlberg, C., Shneiderman, B.: Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In: ACM CHI Conference on Human Factors in Computing Systems, pp. 313–317. ACM Press, New York (1994)
16. Mueller, F., Lockerd, A.: Cheese: tracking mouse movement activity on websites, a tool for user modeling. In: CHI 2001: CHI 2001 extended abstracts on Human factors in computing systems, pp. 279–280. ACM, New York (2001)
17. Costagliola, G., Fuccella, V., Giordano, M., Polese, G.: Monitoring online tests through data visualization. IEEE Transactions on Knowledge and Data Engineering (2009 - to be printed)
18. Green, T.R.G., Petre, M.: Usability analysis of visual programming environments: A cognitive dimensions framework. Journal of Visual Languages and Computing 7, 131–174 (1996)