Abstract—This paper presents a semi-automatic approach to Web applications design evolution which leverages the Ubiquitous Web Applications (UWA) design framework, a methodology and a set of models and tools for the user-centered design of multi-channels and context-aware Web applications. The approach is based on a two-step redesign process: first a semi-automatic reverse modeling phase analyzes the html pages of the application front-end to abstract a model of the “as-is” design, according to the UWA formalism; second, a forward design phase starts from the recovered models and the (new) requirements available for the application to identify lacks and opportunities of improvements in the “as-is” design and produce the “to-be” version of it. The reverse modeling phase applies clustering and clone detection techniques and is supported by an Eclipse IDE environment. The forward design phase is supported by a set of UWA modeling tools which are built on top of the Eclipse Graphical Editing Framework (GEF) and of the Eclipse Graphical Modeling Framework (GMF) and that allow developers to evolve the recovered models. The results from a concrete case study to assess the validity of the redesign approach are also presented and discussed.

Keywords: Web application redesign, Web systems evolution, UWA, clustering, clone detection, Eclipse, GEF, GMF

I. INTRODUCTION

In the last ten years, the development, diffusion, usage, and popularity of Web applications have grown exponentially. Web applications have become the underlying engine of many e-businesses, including e-commerce, e-government, and services such as news and data provision. The complexity of Web applications has increased accordingly to the sophistication of the functionalities they provide.

As a consequence of this trend and due to the continuous change of their requirements, Web applications are subject to continuous maintenance and evolution. Often, modifications are perfective, adaptive, and evolutive maintenance operations for the adoption or the integration of new technologies. As such, most of the required modifications do not need to alter the existing functional behavior or business rules of the Web application, nor the way contents and services are presented to users, but just the operational or technological structure used for its execution. In these cases, the modifications entail mainly the implementation layer, rather than the conceptual design models describing the domain objects and services.

In other cases, evolution is not driven by new technology, but by the need/opportunity to improve aspects characterizing Web applications and influencing their external quality, such as their usability. Namely, three of these aspects are Contents, Navigation and Presentation [10]. The addition of new content types and the extension of existing ones, the addition of new access structures to contents and the modification of existing ones, the enhancement of the navigation structure with new navigation paths, and the improvement of the application user interface, are interventions that impact on the conceptual design of the application and that can be profitably analyzed and represented by means of suitable Web application design models.

The availability of up-to-date instances of such models has a key role to successfully evolve a Web application in response to new requirements. Unfortunately, due to the short time-to-market that often constrains the development and maintenance of Web applications, such documentation is often lacking. This causes maintenance and evolution to become difficult and risky tasks, potentially compromising the effectiveness and correctness of the whole system.

The usage of techniques and tools to recover such models, when lacking, is as useful as needed.

This paper presents an approach for Web applications design evolution based on the Ubiquitous Web Applications (UWA) design framework [19], a framework providing a methodology and a set of models and modeling tools for the user-centered conceptual design of ubiquitous Web applications. The approach is based on a two-steps redesign process which first recovers the UWA models representing the design “as-is” of the application, and then uses these models to reason about needs and opportunities for improvements, and to produce the model of the design “to-be”. These models, additionally to represent an up-to-date documentation artifact for the application, can be used as a reference by developers to implement required changes. Furthermore, they can be used as a starting point of a model-driven development process based on UWA, which would lead to a new implementation of the application [7].

The rest of the paper is organized as follows. Section II discusses some of the works related to ours. Section III describes the overall redesign process and provides some details on the underlying techniques and technologies for both the reverse and the forward design phases. Section IV presents the set of tools supporting the approach and discusses their functionalities. A practical application of the approach to redesign a real-world Web application is reported in Section V. Finally, Section VI concludes the paper by summarizing our contribution and announcing future work we are pursuing.

II. RELATED WORK

In the last years, several approaches have been proposed for the reverse engineering of Web applications. They differ
in the aspects they focus on, the level of abstraction of the recovered information and the formalism they adopt to represent it. The works presented in [13], [8], [15] focus on recovering an architectural view of the Web application depicting its components (i.e., pages, page components, etc.) and the relationships among them at different levels of detail. In [12], an approach for abstracting a description of the functional requirements implemented by the Web application is proposed. UML use case diagrams are used to represent the abstracted functional requirements. UML diagrams are also used in [21] to depict the static, dynamic and behavioural aspects of the analyzed Web application. Approaches and underlying techniques to abstract a model of the business processes implemented by a Web application are presented in [17] and in [9]. The VAQUISTA [20] system, by Vanderdonckt et al., allows the reverse engineering of the presentation model of a Web page, in order to migrate it to another environment. The TERESA tool presented in [14] produces a task-oriented model of a Web application by source code static analysis, where each task represents single page functions triggered by user requests. The resulting model is suitable for assessing Web application usability, or for tracing the profile of the users of the analyzed Web application. The reverse engineering phase of our redesign approach differs from the works cited in this section and others proposed in literature mainly because it refers to a robust and complete methodology specific for the conceptual design of Web applications. As such, the recovered models feature a user-centered perspective on the analyzed application. To the best of our knowledge, no other work deals with the recovering of such user-centered conceptual models. Moreover, being our approach based on client-side source code analysis, it is applicable to any Web application producing HTML pages as front-end, regardless of the technologies used server-side.

III. THE REDESIGN PROCESS

The proposed redesign process consists of two main phases: a reverse engineering phase, and a forward model-driven design phase. The first phase refers to the UWA Web applications design methodology and to a number of code analysis techniques (including clone detection and clustering techniques) to abstract the user-centered conceptual model of the application to redesign. The second phase takes as input the UWA model recovered by the first phase and enables refining and evolving it, e.g., to respond to new application requirements. Tool support is provided for both the two phases which are semi-automatic and require the user intervention mostly to drive the different process activities and validate the produced results.
In the following of this section we first introduce the UWA methodology and, more specifically, its design models; then we describe the two phases in which the redesign process is articulated.

A. The UWA Web Design Methodology

The UWA design framework includes a complete design methodology and a set of models and modeling tools for the user-centered conceptual design of data and operation intensive ubiquitous (i.e., multi-channel, multi-user and generally context-aware) Web applications [19].

Similarly to other well known web engineering methods proposed in the literature, such as OOHDM [16], WebML [5] and UWE [11], UWA specifies the design of a Web application by means of three main models: the Information Model (a.k.a., content or domain model), the Navigation Model, and the Presentation Model [18]. Additional models proposed by UWA include: the Transaction Model, which models the business processes the application is intended to support; the Operation Model, which models how the application will provide to its users; the Customization Model, which specifies, by means of customization rules, how the application will adapt to different usage contexts.

The UWA information model comprises two sub-models: the Hyperbase model and the Access Structures model. The hyperbase model describes the contents of the applications in terms of base information classes (Entities), their structure (Components and Slots) and their relationships (Semantic Associations). The Access Structures model defines subsets (Collections) of the application contents, each based on a selection criterion derived from a specific information access user goal. The UWA navigation model assembles elementary information elements (slots from one or more entities, association centers and collection centers) into reusable units of consumption (Navigation Nodes) and defines navigation contexts (Navigation Clusters) by grouping nodes and defining navigation paths through them (using Navigation Links). Finally, the UWA presentation model specifies how the application is organized in terms of pages, which are the components of each page (Publishing Sections and Publishing Units), and which node is published in each publishing unit.

Each UWA model, indeed, specifies a specific aspect (e.g., navigation) of the designing application from the perspective of the final user. As a consequence, the UWA model of a Web application describes how the application will be perceived by its users, rather than how it is implemented. Different models of the same kind (e.g., different navigation models) can be drawn to define how the application will appear and will behave for different user types. In the above sense, UWA is a user-centered design methodology, and so are its models.

Entities, entity components, semantic associations, collections, navigation nodes, navigation clusters, pages, sections, publishing units and other UWA modeling primitives can be “typed” (the most common case), thus representing classes of objects, or “untyped”, thus representing singletons.

An example of UWA model, including a portion of the Information, Navigation and Publishing models is reported in Figure 4 and 5.

B. The Reverse Engineering Phase

Aiming at describing a Web application at a high level of abstraction and from a user-centered perspective, UWA models are independent of any technology chosen to implement the application. This characteristic makes UWA models suitable to represent potentially any Web application producing HTML pages as front-end. Moving from the above consideration,
we have developed a semi-automated reverse engineering approach which is able to abstract UWA models from existing web applications [4][18]. Here we shortly synthesize this process and the underlying techniques while more details can be found in [4].

The reverse engineering process is made up of five steps which enable the recovery of the following UWA models:
- UWA Information model
- UWA Navigation model
- UWA Publishing model

Figure 1 shows the activity diagram modeling the defined reverse engineering process which has to be carried out on a significant amount of client side HTML pages of the application, to be downloaded from by means of a Web crawler.

1) UWA Information Model Abstraction: The UWA Information model of the considered application is recovered by analyzing its pages to abstract Entities, Semantic Association and Collections.

The identification of UWA Entities is carried out by searching for groups of related attributes (we refer to these attributes as to keywords) in the client-side HTML pages (static and dynamically generated) of the WA. A group of keywords involved in the same user input or output operation and included in the same HTML form or output report is considered as a possible group of Slots characterizing a UWA Entity. The rationale behind this assertion is that the set of data items that a user enters into an input form, or that are shown to a user by an output report, usually represents a concept of interest for the user in the domain of the application. Similar considerations apply to groups of keywords characterizing a set of cloned client pages, i.e. a group of client pages characterized by the same HTML control structure but different content. In this case keywords can be identified by considering labels associated to content items (such as text, images, multimedia objects, etc.), text appearing in table headings, titles appearing in page sections, etc. From each group of cloned client pages a HTML page Template is produced. This template has the same control component and the same set of keywords that are common to all the pages in the set of cloned pages. Each identified keyword is candidate to be a UWA Slot and the keywords in a group are candidate to be an Entity Component. Edit distance metrics and clustering techniques have been defined to identify groups of cloned pages and extract from them associated groups of keywords. A validation phase is manually carried over the automatically identified groups of keywords to finally obtain a validated set of UWA Entities associated to the considered application.

A candidate Semantic Association is assumed to exist between pairs of Entities having some Slots in common. If different Entities are shown in the same HTML page, a candidate Association between them is also considered to exist. Semantic Associations are also derived from hyperlinks connecting pages showing different Entities mainly when a Slot is used as an anchor to set the hyperlink. Similarly to candidate Entities, candidate Associations automatically found in this step have to be validated by a human expert knowledgeable of the application domain.

The identification of UWA Collections is based on the ways they are usually implemented in a WA. These include: (i) the usage of a table where each row reports a different instance of a given Entity or Association; (ii) a list of hyperlinks pointing to pages showing different instances of the same Entity. As for Entities and Semantic Associations, the automatically recovered UWA Collections will undergo to a validation phase conducted by a human expert knowledgeable of the application domain.

2) UWA Navigation Model Abstraction: The recovery of the UWA Navigation Model is carried out by identifying Nodes and Clusters for the analyzed application. Nodes are identified by associating them to structural sections in the pages of the application, displaying the requiring information from/to the user. The client pages related to Entities, Associations, and Collections are selected and analyzed to: (i) identify which attributes of each Entities, Associations, or Collections are referred in the page; (ii) associate a Node to each group of attributes; (iii) identify hyperlinks connecting Nodes in the same page or in different pages. Links between nodes are used to identify Navigation Clusters. A list of Nodes and their organization into Clusters is the result of this step. Each Node and each Cluster is assigned a unique name derived from the elements of the Information Model they are associated to.

3) UWA Publishing Model Abstraction: The UWA Publishing Model of the analyzed application is abstracted by identifying Publishing Pages, Publishing Sections and Publishing Units (PU) from the set of templates obtained during the phase of Information Model recovery. A Page is associated to each template contributing to the identification of at least an Entity. To identify sections and PUs associated to each page we assume that a Section includes only one PU and associate a PU to each of the Nodes recovered in the Navigation Model. By tracing the association between Nodes and templates it is possible to associate PU to Pages.

C. The Forward Design Phase

The UWA model recovered with the reverse engineering phase provides a representation of the design “as-is” of the analyzed application. By examining this model and by considering the (eventually new) requirements available for the application, as well as the design guidelines provided by the UWA design methodology, the analyst can:
- identify lacks and weaknesses of the current design;
- define changes in order to overcome them;
- evolve the design to meet new or changed requirements.

An example of such analysis and of the evolution interventions that can be applied can be found in the case study section of the paper.

The data gathered during the reverse engineering phase by the RE-UWA tool platform are exported so to become a well-formed instance of the UWA MOF metamodel. The UWA model obtained in this way is then imported in the UWA graphical editor described in Section IV. Figure 4 and 5 show
excerpts of the UWA model recovered for the Exibart.com web site considered in our case study.

By means of the UWA modeling tools, we can introduce changes in the recovered model to overcome lacks or weaknesses identified in the “as-is” design of the application and we can evolve it to better meet available requirements.

The new model can be used as a reference to implement changes in the application. As an additional option, the new model can be used as a starting point to re-implement the application by applying the UWA model-driven Web application development approach detailed in [7].

IV. TOOL SUPPORT

A. The RE-UWA Tool Platform

Figure 2 (a) shows the architecture of the RE-UWA Tool Platform (RTP). At the lowest level of the architecture is the RTP Core layer that introduces project integration providing builders aware of UWA resources and a project nature enabling RE-UWA process workflow for Eclipse WTP projects. In this layer there are also basic services for the entire RTP: HTML/XML parsers, along with similarity distance calculators between HTML documents and core platform services.

The platform services implement the logic to import the pages of the Web application into a UWA project. Such import phase extracts structural information about: (i) the downloaded client pages; (ii) the inner components of each page (e.g. forms, scripts module, frame, applet, etc.); (iii) the hyperlinks connecting the pages.

The extracted information are stored into a repository located in the analyzed project. A clone detector module is used to perform static analysis on the HTML client Web pages of the application to identify pages that are clones. The clone detector component traverses HTML/XML DOMs to generate distance matrices for the HTML pages under analysis. This component can be configured by independent modules that gain access to the DOMs of WA pages to calculate the distance matrix with several distance algorithms. Currently, as discussed in Section III, two distances are supported: the edit distance and a maximum sub-tree matching distance. The data extracted are made accessible to the entire RTP environment.

The RTP Process layer implements the process logic: it is based on a workflow engine that follows the RE-UWA process specification. For each step of the process there is a component implementing it. The engine takes the process instance and transfers the control between the steps as specified in the process definition.

The process is structured as a direct graph in which there are several kind of nodes and edges. Nodes can be simple nodes or composite ones with an inner structure. Simple nodes can be process (executing recovering process logic) or predicate nodes (to structure the control and data flow). Composite nodes can be of several types depending on the policy of execution of inner nodes (i.e., all nodes must be executed; only one must be executed; any of inner nodes can be executed). Edges are of different types according to the needs for interaction on the transition and on the routing policies (auto or manual routing).

The software components participate to the framework by inheritance and composition: they can be added, removed or modified in flexible ways. Each process has a customizable configuration phase where control and data flow dependencies, among the steps involved in the process, can be specified.

1) RTP IDE Layer: This layer implements the presentation layer that allows the interaction with users to drive the process execution. It is structured as a set of Eclipse editors and views that interact with the engine and the concrete components. It allows the analyst to execute the step logic providing the needed and related information to support her/his choices. The RTP IDE layer introduces the following perspectives each one related to the recovering of a well defined portion of the UWA model:

- **Forms and Clones Perspective** - This perspective contains all the views related to HTML page clustering, templates generation and group of keywords extraction and validation. In this perspective there are also some editors defined to handle the recovered elements.
- **Entities and Associations Perspective** - This perspective contains all the views related to UWA Entities and Semantic Associations. Entities are added to this view from: (i) the output of the algorithm discussed in Section III; (ii) the Entities specified by performing the semi-automatic analysis using the Web Page Designer (WPD) editor embedded in Eclipse. The validated and refined Entities and Associations can be finally saved into the internal repository.
- **Collections Perspective** - The Collections Perspective groups together the views used to drive the recovering of collections and their centers. For each identified Collection, the Collection view shows the pages containing it. By using the collection editor, the analyst can validate the identified Collections and refine them.
- **Navigation Model Views** - The Navigation views are a set of views used by the analyst to generate and to browse the UWA Navigation model from the information regarding Entities, Semantic Associations and Collections stored in the RTP repository. Clusters and related nodes, along with links among them and the attributes, are also showed.

Figure 3 shows a screenshot of the RTP tool in the perspective for the recovery of UWA Entities and Associations captured during the analysis of the Web site Exibart.com.

B. The UWA Modeling Tool

Starting from the UWA conceptual meta-model, created with the Eclipse Modeling Framework (EMF) [3] tool and the Ecore meta-language, a visual editor has been developed to graphically instantiate conceptual models and in order to replace the default tree editor offered by the EMF tool. For this purpose we have tested EMF in combination with the Graphical Editing Framework (GEF) [1], which aims to facilitate the development of customizable graphical editors starting from existing meta-models. Unfortunately, these two frameworks are not born to cooperate and there are some incompatibilities that make difficult their integration. Another approach for developing model-based graphical editors (as Eclipse plug-ins) is provided by the Graphical Modeling Framework (GMF) [2]
tool, which can be considered the merging of EMF with GEF. While GEF requires a substantial effort in coding, GMF has the ambitious goal of reducing or removing the writing phase of the program, by replacing it with the definition of a series of intermediate models to automatically generate the code of the editor. For this reason we have decided to choose this second way instead of the combination of EMF with GEF. The GMF, with what we might call “model-based programming”, is certainly the most promising idea to quickly create feature-rich graphical editors starting from models.

As shown in Figure 2 (b), to define the UWA meta-model we used EMF and the Ecore meta-language to support the meta-modeling process, while for realizing the UWA editor starting from the meta-model, we used GMF. The output of this phase has been a graphical editor with the features of drag and drop, copy and paste, undo and redo that allows to draw UWA diagrams using the tools palette and to connect, re-size or move the model elements.

V. CASE STUDY

In order to validate our redesign approach we have applied it to a real world Web site. The selected application is Exibart.com, an art news portal, reachable at http://www.exibart.com, which provides information on art exhibitions, events, performances, with reviews and comments by art curators and visitors.
TABLE IV
IDENTIFIED UWA COLLECTIONS

<table>
<thead>
<tr>
<th>Identified Collections</th>
<th>Slots of Collection Centre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related Authors</td>
<td>Author's name</td>
</tr>
<tr>
<td>Comments</td>
<td>Author's name</td>
</tr>
<tr>
<td>Last Comments</td>
<td>Author's name</td>
</tr>
<tr>
<td>Events of the Day</td>
<td>Event's name</td>
</tr>
<tr>
<td>Events of the Day</td>
<td>Exhibition area name</td>
</tr>
<tr>
<td>News</td>
<td>Title</td>
</tr>
<tr>
<td>Latest News</td>
<td>Title, Place, Author's name, Text, Image</td>
</tr>
<tr>
<td>News of an Author</td>
<td>Date, Title, Place</td>
</tr>
<tr>
<td>Most clicked News</td>
<td>Title, Place</td>
</tr>
<tr>
<td>Artist Parade</td>
<td>Artist's name</td>
</tr>
<tr>
<td>Curator Parade</td>
<td>Curator's name</td>
</tr>
</tbody>
</table>

A. Reverse Modeling Exibart.com

An instance of the Exibart.com Web application was mirrored and around 9000 pages were selected. A preliminary similarity analysis was performed on these pages and 2500 of them were selected to cover the different sections of the Web site, with an average of 70 pages for section. Clone analysis and clustering techniques were used in order to group similar pages.

The reverse engineering phase was executed at increasing clustering thresholds to obtain best possible results. For each threshold value, precision and recall were calculated. Degenerate clusters containing a single page or useless pages were discarded before the model abstraction steps. From

Fig. 4. A screenshot of the UWA editor showing the UWA hyperbase model recovered for Exibart.com

Fig. 6. Precision and Recall for the recovered UWA Entities for exibart.com
each cluster, a template was generated and for each valid template, a group of attributes containing potential keywords was extracted using the content extraction algorithm detailed in [4]. These groups were then validated by the human expert using the RE-UWA tool perspective that allows discarding invalid groups and refining the valid ones (by means of merge, split and modify operations). Table I reports, for each threshold, the number of total groups, the valid groups and the rejected ones. The computed groups represent the candidate UWA Entities; the abstraction algorithm defined in [6] is used to recover the actual UWA Entities.

Table II reports the UWA Entities identified by the approach, for the best achieved clustering threshold. For two Entities there was no threshold capable of identifying them, and hence they required manual identification. This was due to the lack of keywords in the pages containing the Entities: in these cases templates were empty (and hence discarded in the previous step) or contained invalid keywords and hence originated groups discarded later during validation.

After the abstraction of UWA Entities, UWA Semantic Association were identified among them. This step produced the 13 Semantic Associations showed in the matrix reported in Table III (where the character ’m’ is used for manually recovered associations).

Using data recovered on UWA Entities, UWA Semantic Association and the clustered pages, UWA Navigation Nodes and UWA Navigation Clusters were recovered.

For each UWA Entity a UWA Navigation Cluster was generated, by considering the nodes generated from clusters of Web pages containing that UWA Entity. The same was done for UWA Collections.

From the Navigation Clusters of the Navigation Model, the UWA Publishing Model was recovered as described in Section III-B.

Figure 4 reports an excerpt of the UWA Information Model recovered for Exibart.com, particular the Hyperbase Model, showing the recovered Entities with their internal Components and Slots. Figure 5 shows a subset of the recovered Semantic
TABLE II
UWA ENTITIES IDENTIFIED BY THE APPROACH

<table>
<thead>
<tr>
<th>Threshold</th>
<th>#Templates</th>
<th>Author</th>
<th>Comment</th>
<th>Event</th>
<th>News</th>
<th>Person</th>
<th>Exhibition Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.100</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.080</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.060</td>
<td>40</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.040</td>
<td>55</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.030</td>
<td>68</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.020</td>
<td>68</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.010</td>
<td>69</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.005</td>
<td>77</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>0.003</td>
<td>91</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>m</td>
<td>m</td>
<td>m</td>
</tr>
</tbody>
</table>

Associations with their Centers, a subset of the recovered Collections, and excerpts of the Navigation and Publishing Models.

1) Validation of results: In order to validate the results, we calculated precision and recall for each clustering threshold. Precision was calculated using the valid groups extracted from templates with respect to the total number of generated groups. Conversely, recall was calculated by manually inspecting the application in order to compute the ratio between the number of identified entities and the total number of the entities defined in the application.

Figure 6 shows the precision and recall for all the different clustering thresholds. As clone detection technique, we used the edit distance applied to the linearized DOM tree of the pages as specified in [4]. As we can see, with a threshold of 0.08 only 1 entity out of 6 were correctly identified (with a low precision, around 0.34). Decreasing the threshold to 0.06, 3 entities were identified giving a recall of 0.5 and with a better and acceptable precision (p=0.43). The best values were obtained at a threshold of 0.03; in this case 4 entities were automatically recovered (improving recall to r=0.57) but with a much better precision (p=0.61). Continuing to decrease the threshold did not allow the identification of more entities since the remaining three were in pages in which no keywords are used and hence a completely manual identification is needed. Due to space restriction we only report precision and recall for Entities. However, we have obtained comparable results also for Semantic Associations, Collections and for Navigation and Publishing Models.

B. Improving the Design of Exibart.com

Figure 4 shows the UWA Hyperbase Model recovered for the Exibart.com Web site. By analyzing this model, we can identify a number of lacks and derive possible improvements. First of all the Entities “Artist”, “Author” and “Art_Curator” appear very poor in information and could be enriched. As an example, it would be useful for the visitor of the site, to read artist’s biographies and to know authors’ and art curators’ curricula. Secondly, the Semantic Association existing between “Author” and “News” is missing the counterpart enabling navigating from a given news to the page of its author.

Let’s now consider Figure 5 which shows a detailed view of some of the Semantic Associations and Collections diagrams recovered for the same site, and a portion of the Navigation and Publishing Models. In this diagrams, Semantic Associations and Collections are presented with their centers and included slots. By analyzing these centers, the analyst can evaluate if a Collection or an Association is defined properly or if it can be better presented. In fact, UWA design guidelines suggest that slots to be included in collections and associations centers should be chosen based on their purpose to “announce” the entity they will link to. As an example, the collection of “Most clicked news” presented in the home page of Exibart.com lists news by means of their titles, while it would be useful to also show their publication date, a short text, and their authors. In this way the user of the Web site can take a more conscientious decision on which news might be of his/her interest and click to visit it.

Moving onto the Navigation Model, among other aspects, a UWA Navigation Cluster specifies the navigation pattern used within the Cluster to navigate between Nodes included in it. Typical navigation patterns are Index, All-All, Guided tour, Index/guided tour. All of the collections identified in the Exibart.com Web site can be navigated by index, i.e., navigating from the center of the collection to its members. This is represented by the recovered UWA Navigation Model. The collection of events that took place in an exhibition area should instead support a guided tour navigation pattern, so to enable the user navigating from an event to the successive, in a temporal order.

Finally, the analysis of the UWA Publishing Model can highlight Pages or Sections of the site that use a layout different from the rest, or publishing units which are redundantly repeated throughout the site. In Exibart.com, this was the case of the Publishing Unit “Speed News”, which is used to publish the collection of latest news both in the home page but also in all the other pages of the site.

C. Discussion of Results and Limits of the Approach

The results obtained in the case study involving the Exibart.com Web site indicate that our redesign approach is feasible and valid. In fact, it enables taking the benefits offered by a design methodology specifically suited for Web applications design, such as UWA, on existing Web applications. Applying our redesign approach makes it possible (i) recovering a model of the current design of the application, which might
be useful for redocumentation purposes, (ii) identifying lacks and possible solutions, and (iii) evolving the current design in order to meet new or changed requirements. Additionally, as UWA provides tools for the model-driven development of the designed applications [7], our redesign process can be followed by an implementation phase, which lead to the reengineering of the application.

The case study has also highlighted some limits of the approach, mostly related to the reverse engineering phase. A limit that we identified is related to the impossibility to identify UWA Entities in an automatic way when the Web site does not make use of keywords associated to contents. In this case, Entities must be identifies manually, thus increasing the effort required to perform the reverse engineering step. Another limit is related to the identification UWA Entity Components: the slots identified for an Entity are organized into a single component, which requires more effort in refining the model in the forward design phase.

VI. CONCLUSIONS

This paper presents an approach and a supporting tool platform for the reverse design and evolution of existing Web applications using the UWA design framework and related models.

The approach relies on a semi-automatic reverse engineering phase that exploits clustering and clone detection techniques to abstract UWA conceptual models from the existing application by analyzing the client-side HTML pages. The recovered models, additionally to represent an up-to-date documentation of the design “as is” of the application, are then used to identify weaknesses and propose solutions to overcome them, as well as to meet new requirements.

The proposed approach enables taking advantage from applying a framework specifically suited for the user-centered design of ubiquitous Web applications, to redesign existing Web applications.

The conducted case study proved the feasibility and effectiveness of the approach which was able to identify a number of lacks in the analyzed Web application conceptual design and to suggest ways to overcome them.

Future works will be devoted to improve the reverse engineering phase in order to recover also the UWA Operation and Transaction models, useful to represent the business process aspects of the analyzed applications. We also aim to conduct additional case studies on a wider set of Web applications. Finally, we are also working to further extend the UWA modeling tools with new features, by exploiting the new potentials that the GMF framework will provide in the near future. One of these extensions concerns the possibility to organize the whole UWA model of an application in separated but related GMF diagrams so to ease their editing and modification.

REFERENCES
