What Kind of Verification of Formal Navigation Modelling for Reliable and Usable Web Applications?

Marco Winckler\textsuperscript{1} Eric Barboni\textsuperscript{2} Philippe Palanque\textsuperscript{3} Christelle Farenc\textsuperscript{4}

LIIHS-IRIT
University Paul Sabatier
Toulouse, France

Abstract

In this paper we introduce briefly a notation dedicated to model navigation of Web applications and we discuss some strategies to assess the usability over navigation models built with such a notation. Our aim with this kind of evaluation is to ensure (prior to implementation) that important users tasks can (or cannot) be performed using the system.

Keywords: Model-based Web development, Model-based verification, Formal description techniques, Usability evaluation.

1 Navigation modelling with the SWC notation

Model-based design is a relatively recent field over the Web but it is growing fast due the demanding need of modelling support to build more and more complex Web applications [6]. However, even if appropriate modelling techniques exist, the use of modelling methods alone it is not enough to ensure the usability and reliability of Web applications. Usability evaluation is not

\textsuperscript{1} Email: winckler@irit.fr
\textsuperscript{2} Email: barboni@irit.fr
\textsuperscript{3} Email: palanque@irit.fr
\textsuperscript{4} Email: farenc@irit.fr
only important for identifying problems in the early phases of the development process but also for managing the frequent updates performed over web applications.

In order to overcome the limitations of informal models, we have proposed the StateWebCharts (SWC)\cite{9} notation for describing navigation over Web application. SWC is a formal notation that extends StateCharts \cite{5} by adding appropriate semantics for states and transitions, includes notions like dialogue initiative control and client and transient activities. More generally, Web pages (static or dynamically generated) are associated to SWC states. Thus, each state describes which objects users can see when navigating over the applications. Links are represented by transitions between states. Data can be transferred from a state to another through the transitions. Persistent data can be held by a special kind of state called transient state. The operational semantic for a SWC state is: current states and their containers are visible for users while non-current states are hidden. Users can only navigate outgoing relationships (represented by the means of transitions in the model) from current states. When a user selects a transition the system leaves the source state which becomes inactive letting the target state to be the next active state in the configuration.

Figure 1 provides view at a glance of the SWC notation presenting the navigation modelling a small search engine. The decoration starting with a black dot over the state ”S1” indicates the initial state. This concept of initial state as well as many others constructs are a heritage of StateCharts. External states (e.g. ”S5”) represent relationships to states that are out of the control of the designers. External states are necessary because links towards other Web applications should be considered as part of the design. Similarly, SWC provides other types of states to represent the possible inclusion of dynamic content containing navigation path that cannot be determined at the design time (e.g. ”S3”). The state ”S2” represents transient states which usually represent the system engine. Dashed arrows (e.g. ”t3”, ”t2”) indicate transitions triggered by the system while continuous arrows (e.g. ”t1”, ”t4”, ”t5”) identify transitions triggered by user events. A detailed description of the SWC notation can be found in \cite{9}.
2 Strategies for model-based evaluation of navigation

In the last year, several methods have been proposed for supporting model-based evaluation such as model-checking [1] and inspection of system properties [2]. In this section we present briefly 4 strategies for assessing usability over navigation models: a) static verification; b) dynamic verification; c) verification of system properties; and d) assessment of navigation model using user scenarios. We aim at discussing the potential benefits of each method with respect to the development of Web applications.

Static verification refers to analysis without model execution. Several methods have been proposed as described in [2] for supporting model-based evaluation such as model checking (typically used for software testing, it is used to check intrinsic elements of a model) [7]. This category covers the following issues: a) Model consistency; b) Unreachable states; c) Dangling links; d) Redundancy of links; e) Unreachable paths; f) Shortest path; g) Compulsory path; and h) Data dependency.

The principle of dynamic verification is that the model must be executed or simulated in order to perform the verification. The simulation, step by step under the control of the designer, can be used as a kind of walkthrough method for inspecting both navigation specifications and their corresponding Web applications. In this case, the designer ‘navigates’ the model as a real user could do. This task can be eased by the appropriated tool support. Walk-through over model is not the unique way for performing dynamic verification since the navigation models can also be accessed to measure the coverage of a given test suit or a collection of test cases. The test suit consists of a set of directives which are used by automated tools for inspecting the model [1]; for example: a) every state is visited at least once in some test case; b) every relationship from every state is followed at least once; every path in the site is followed in at least one test case.
The verification of properties is made up by inspecting the models in order to verify if the model is compatible or not with some predefined behavioural rules (as we only consider in this paper properties concerned by navigation) called properties. They can be expressed by using Temporal Logic [3] or any other abstract declarative formal description technique. Properties can be very generic/abstract or very concrete such as ”do not allow more than seven relationships going out from a state”. This kind of property might be verified automatically over the model and even embedded into editor tools but this heavily depends on the expressive power of the notation and the complexity of the models. Until now, only few works [4] have been devoted to verify ergonomic rules as properties of models.

Assessment of navigation model using user scenarios aims at exploiting two complementary models and to cross-check their compatibility. Task models are aimed at describing not only how users perform their task but also when and why those tasks are performed (in order to achieve which goal). Task models are typically a result of a detailed specification of functional requirements of the interface describing user tasks with the application. The basis of such an assessment is to extract scenarios from tasks models and to play them over navigation models [10]. Contrarily to other works that analyze navigation paths, this procedure allows designers comparing navigation paths to real tasks, which is supposed to give deeper insights about the usability of the web application’s user interface.

3 Discussion and future work

As discussed by Campos and Harrison [2] and Farenc [4] it is possible to associate the results of model-based evaluations and the principles of usability for interactive system. The verification of static properties of models has a long tradition in the discipline of Formal Methods and many tools exist for this purpose. The dynamic verification of model by simulation has also been used to evaluate the navigation of web applications [7][8].

Currently, the main drawbacks with model-based evaluation of navigation models concern the dynamic content generation and the identification of user goal of navigation. The automatic generation of content might include new links that can modify the usability predicted by evaluations performed over navigation models. The goal-driven evaluation cannot be solved just using navigation models because we cannot match a navigation path to a single user goal. In fact, user can often follow many navigation paths to accomplish the same goal. Goal-driven evaluation over model is made possible if we combine synergistically scenarios extracted from task models and navigation models,
as we have demonstrated in [10].

One of the main questions we have addressed in this ongoing research is to identify which elements of modelling methods can be used for evaluation purposes. In that sense, we have proposed a particular semantic for SWC elements that allows, for example, the identification of the content generation by the means of the so-called dynamic states and a formal representation of the agent causing changes in the system states which is essential to evaluate the control of the dialog.

The SWC notation also provides all the set of basic information concerning states, relationships and events which is required to deal with the evaluation described above. The edition of SWC models is supported by the SWCEditor [8] which also allows the simulation and verification of models. Our ongoing work consists in developing and integrating analysis tools to our prototype in order to support a cumbersome and resource demanding manual process.

References


