Dynamic web-components and web environment behavior analysis

Suvorov V.[1]

[1]Saint- Petersburg State University, Math and Mechanics faculty

A problem of automated interaction with complicated website at a user level is considered. The general case of playing online game is considered as a model of any site usage. Three functional components – the user actions, the site logic and other parties' actions are distinguished. A method of reconstructing site components, their behavior and interconnectivity is suggested. Practical results of using the method are shown - playing online game and getting components from broadcast site.

***Index Terms*— website automation, web-component analysis, website behavior analysis, website integration**

I. Preface

Nowadays internet applications are widespread. It is declared that the era of Web 2.0 technologies came up – that means the idea of the proliferation of interconnectivity and interactivity of web-delivered content [1]. Sites incorporate new technologies of creating dynamic web pages and complicated user interaction scenarios. There are various technologies of creating dynamic content and it is not so easy to unite them or use some part of one site in the other as it was in pure HTML –era. As content is generated at the server side, it is not enough to send URL as request and analyze the reply of the server, as it is just a static stamp of some particular case.

As Web becomes some media for applications, the problem of non-transparency occurs. Yesterday one could just analyze the source code of webpage and understand the behavior – now it is pretty hard and non-trivial, some parts comes as black –boxes (for example flash container)

There exist tools that help to understand site structure, for example, Firebug for Mozilla but this tool does not allow automating information retrieval. The search robots distinct text and links and not components or behavior. There are cases when automation of using site is needed

Use some component from third-party site in your own

Automate actions on third-party site (post blog, play online game, etc.)

These tasks are usually solved locally that means writing a specific script for specific task. An example is various Greasemonkey scripts. In the paper a method of reconstructing site components, their behavior and interconnectivity is suggested so that then it will be possible to provide a visual environment of creating automation scenarios and solve the problem of components' reuse. The general case of playing online game is considered as a model of any site usage. The model is robust as it has three components - the user actions, the site logic, and other parties' actions.

Some difficulties to be faced

The plain HTML pages dissolve in the sea of the dynamic content namely PHP, ASP, Flash and others. The AJAX technology is also frequently used. One can name lots of other technologies used in creating web pages but the idea is that client –server communication is not based solely on hyperlink clicking but on different other scenarios. It results in a dynamically updated webpage. Dynamic content is generated either on a server side, e.g. PHP or on the client side e.g. JavaScript. For the moment let us neglect the problem of indexing such dynamic pages - that problem the Google successfully solves, but let us face other problem of linking different web applications. That seems easy if you want to just take some part of the webpage and encapsulate it in your own. Nevertheless, you can expect some difficulties with AJAX components and find out that as you moved a component it does not work properly for example the authorization procedure is missing. You have to reverse the code, which is trivial for static html and non-trivial for dynamic content and restore the communication model of the component you use.

Another difficulty exists if you want to automate information retrieval from the dynamic webpage. You may need to authorize or to navigate through some links, as the static link not always exists. That difficulty is now faced by search engines - the search robot indexes page but then the user follows the link provided and finds out other content. The solution used is to provide the webpage cached by search bot. This is not the best solution as it makes a security hole as one can have access the information on a particular webpage without required authorization and may have some copy legacy aspects. The search bot indexed the page under some account and cached page. That unintentionally reveals the content of authorized user to unauthorized user or user blocked by IP-filter.

As it seems trivial at the first sight to get some content from a webpage for your own use - it is not, so if the webpage is complex, dynamic, and recent technologies make it even worse. Usually some scripts for particular web content and particular site are written and locally solve the problem of retrieving information.

Not so long ago interactive web-applications were developed. They require the user to stay online for a significant time and provide dynamically changed online content. The example is online games - numerous of them. One of the most well known is Travian. It has more than 1000000 players worldwide and about one-third online at any moment. Another example is stock applications and online broadcasts. These services sometimes provide tools for automation and sometimes not, for example, online broadcasts are not recorded. If it is a big project, then adding some feature by its developers can take a long time. In addition, for any third party is it not easy due to the complexity of webpage structure.

It will be nice if we have a tool that can use for dealing with complex dynamic content – so we can distinguish the dynamic components, analyze them in a simple way and use them for our own needs. For example, we would like to have a bot that will play online game or record interesting broadcasts (the criteria can be based on number of active viewers) or automatically place bets on an auction when it is near the end.

II. A GAME AS A GENERAL MODEL FOR A SITE

Let us construct a natural model of a game. In general, we can describe a strategy game as follows:

Each player initially has some number of resources R (let us say an army is a resource) and buildings B which provide resources in time.

A player can construct buildings by spending resources but new buildings provide resources faster

Resources can be exchanged one for other. Resources can be exchanged between players, so the war is considered as a resource exchange

There exist a set of actions that user can perform, that affect the system state

A simple model we can use is a model with feedback propagating in time

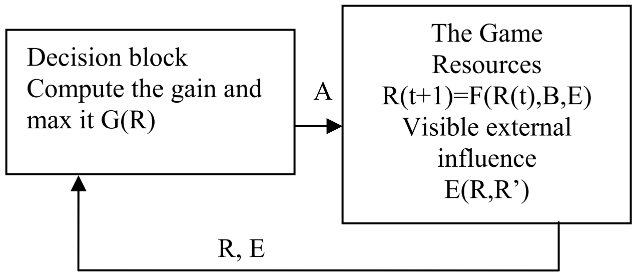

Figure 1. A simple model of the game

E indicates interaction with other players' actions. As we defined that interaction involves only resource exchange so we can say that given R(t) and R'(t) at the time t and applying E we get some resource flow $\Delta R(t+\Delta t)=E(R(t),R'(t))$ and $\Delta R'(t+\Delta t)=E(R(t),R'(t))$ respectfully. In fact we can consider the Nash equilibrium to be reached –so E is predictable and there exist a number of strategies that maximize the gain function. That means the function A(R,E,G) is defined for a chosen strategy G. F is a function that determines resource growth and can be derived from game rules with respect to R and B. As E can be determined either in a solid or in a probabilistic form then F can be determined as well.

That model can be spread on other applications for example in recording online broadcasts the resources will be the number of peers watching, the time of broadcast, disk space.

For example, we would like to watch the most interesting moments – then we should record only most viewed broadcast. (Suppose we record from screen and have a limit of one broadcast a moment)

Why should we use this model? The reason is that we want to deal with dynamic content in a nice way - have nearly all useful properties of the component accessible and automate our actions also in a nice way – just specify rules of the game or even automate learning the rules.

Analyzing communication protocol and modeling communication is considered in [2, 3]. There a finite state machine model is proposed to analyze the behavior and it works well if number of states is finite. In general we can consider states finite but there is a problem of describing such states as initially we do not know the component's algorithm and our goal is not model checking but finding out some relations between actions and resources in time.

So we want an analyzer, which on the input get some URL and maybe some rules (function G(R)) and on the output provide us with analyzed dynamic components and maybe some further rules of intercommunication (F, E)

III. COMPONENT ANALYSIS

First, we need to specify control elements. For that purpose, a simple strategy can be implemented. A sniffer, browser and a program-controlled mouse is needed. Let us consider an action. First, we stay calm and do nothing and record some number of system states. We use screen version of browser to determine if anything changed and detect the regions where picture changed. That way we determine the "noise" generated by counters, banners. Then we try to check points on screen by moving mouse to the point and clicking. Not many points must be checked as the design is user-friendly and controls have reasonable dimensions. If the click was successful –that means something changed on a screen except "noise" then we try to specify the area that changed and sniff packets at the same time. Absence of packets may tell us that the control is a JavaScript and we should iterate through some more so we click around the area that changed or try to input some text – that way we determine the behavior of the control. Presence and type of packets can also give us information about control usage. Next, we have to try to map control into code. That can be done by using Firebug script for Firefox.

Then we should determine the static and dynamic components that are not controls and distinguish parts of the webpage that change by applying a simple differential scheme. That can be done using any client-imitating environment. The environment must fully support HTTP protocol and be able to track AJAX and JavaScript. The easiest way is to implement a sniffer for packets tracking and use custom scripts to control the Mozilla browser. Mozilla is the best choice because it is open source and easy to integrate with. Some part of the mechanism can be written even in grease monkey plug-in using JavaScript. Therefore, we can distinguish the dynamic part.

As a result, a set of dynamic components is created, including controls. If by activating some control, we capture communication we name it a resource and store to the database the content. The content itself can also be static and dynamic – that can be easily checked by series of tests by activating the same control with all possible feasible values.

Every dynamic component even noise is marked as a resource. Resources must have values so we must assign some parameters to components. In general, it is array of strings. The component can change its image, size (shape), text. First, we try with the latter and analyze the corresponding piece of code with regular pattern of readable letters. A dictionary or some method of word recognition is applied at that stage. If the procedure fails, we try to determine size change by differential analysis of series of images of the component and calculating overall estimate. If that approach also fail then we consider set of images and enumerate them in some way.

IV. Behavior analysis

We perform a series of tests and try to find out restrictions and behavior. The user specifies the gain function and the model to use. While testing the gain is maximized by the following strategy:

Do nothing for some time and see what changes. Compute gain and make decision: if nothing changes or gain decreases –do some action, if gain increases do nothing for some time.

Do some action. Compute gain. Wait. Compute change in time and make decision: if nothing changes or gain decreases –do some action, if gain increases do nothing for some time.

In fact the idea is nearly to be Monte-Carlo method but with some little improvements. As a result, we obtain parameters of the behavior model of the site and a strategy of maximizing gain function.

There is still a problem of choosing appropriate model and it will be a further research which ones are better.

V. EXPERIMENTAL RESULTS

The idea was implemented in a computer program. The main features of the program are the following:

- Separate static and dynamic content
- Use only natural user environment – that means HTTP protocol
- Have some feedback mechanism
- In dynamic content distinguish resources and action controls
- Provide some mathematical models for modeling function F
- Determine parameters of the model in a test series.

The tests were provided for the online game Travian. The account was manually created. Then the program worked. First, it used sniffer and applied the differential analysis to HTTP content, so resources were determined. Then by automatic browser, control the program clicked on the webpage and analyze if anything changed except the previously determined resources or if we went to other page by hyperlink. If something changed but we did not go to other page, it tried to see if some data can be input first by analyzing source code and if failed, by trying to "click and input". The number of manipulations (clicks and inputs) was limited by a parameter. The "successful" combination meant resources change or page change and was recordered in database. Grouping was made by the webpage position of the component. The initial step was 10 pixels (between clickable points) and if a successful combination was found for some point, the area of component was determined by repeating the same actions for points nearby. The program distinguished resources components and determined the dependency of resources from time and actions, so that automated building was easily available. The program was also tested at the site smotri.com where the broadcast components and broadcast indexes were distinguished so it made available to construct a page similar to smotri.com broadcast page but without advertisement. Many problems occur in analyzing complex structures, model fitting, and war analyzing -that will be the topic of further research and improvements.

VI. Conclusions

As web becomes more and more sophisticated, the problem of analyzing sites at the functional level will grow. The analysis at the user level provides large flexibility and transparency. A game model is applicable to some even non-game sites. A suggested concept of analyzing components and behavior naturally and empirically proved its value and however many problems emerge they are solvable and need further research.

References

[1] http://en.wikipedia.org/wiki/Web_2.0

[2] Analyzing conversations of web services T Bultan, X Fu et. al. IEEE Internet Computing 2006

[3] Jyotishman Pathak, Samik Basu et al. On Context-Sensitive Substitutability of Web Services In 5th IEEE International Conference on Web Services -2007

[4] ANALYZING USER BEHAVIOR On The Web, Eelco Herder, Ph.D. Thesis, University of Twente, 2006

[5] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008.