Cost and Response Time Simulation
for Web-based Applications on Mobile Channels

Matthias Book, Volker Gruhn, Malte Hülder, André Köhler

Chair of Applied Telematics/e-Business, Dept. of Computer Science, University of Leipzig

Klostergasse 3, 04109 Leipzig, Germany; Phone: +49-341-97-32330

{book, gruhn, huelder, koehler}@ebus.informatik.uni-leipzig.de

Andreas Kriegel

Commerz Business Consulting, Commerzbank AG

Mainzer Landstraße 185, 60327 Frankfurt am Main, Germany; Phone: +49-69-136-41108

andreas.kriegel@commerzbank.com

Abstract

When considering the addition of a mobile presentation channel to an existing web-based application, a key question that has to be answered even before development begins is how the mobile channel's characteristics will impact the user experience and the cost of using the application. If either of these factors is outside acceptable limits, economical considerations may forbid adding the channels, even if it would be feasible from a purely technical perspective. Both of these factors depend considerably on two metrics: The time required to transmit data over the mobile network, and the volume transmitted.

The PETTICOAT method presented in this paper uses the dialog flow model and web server log files of an existing application to identify typical interaction sequences and to compile volume statistics, which are then run through a tool that simulates the volume and time that would be incurred by executing the interaction sequences on a mobile channel. From the simulated volume and time data, we can then calculate the cost of accessing the application on a mobile channel.

Type of submission: *Research paper*
Keywords: *Web engineering, mobile communications, cost estimation*
Topics: *Software quality: measurement; Information management: economics of software quality; Emerging technology: mobile computing*

1. Introduction

As thin client applications, web-based applications have the advantage of independence from the user and his preferred device. Just the existence of a browser and a suitable network connection are needed. Thus, web-based applications seem to be convenient for mobile use. But in hands-on trials of such scenarios, the response time of the application is often notably worse compared to its use in a LAN environment. Furthermore, the communication costs are hard to predict. An organization that plans to provide mobile access to its existing web-based applications for a large group of mobile workers needs detailed information about the usability and estimated cost of the application in a mobile environment *before* investing any effort in building it. Therefore, the expected performance (in terms of response time) as well as the expected cost of the application on different mobile networks need to be quantified at an early stage. With PETTICOAT (**Pe**rformance **T**uning and cos**t** d**isc**overy of m**o**bile web-based **A**pplica**t**ions), we present a method that can be used to address this situation.

The PETTICOAT method can be used by software developers as well as software project managers. After compiling all necessary information, a tool calculates indicators that reveal the application's response time and communication costs in the mobile environment. This way, decisions about the development of a mobile channel for an application can be based on quantitative arguments. If the application is classified as not immediately suitable for mobile use, decision makers can use the detailed results to consider whether it is reasonable to address particular deficits in the application's design revealed by the simulation. This optimization can be conducted for single features or the whole application.

In this paper, we describe how the PETTICOAT method was employed in a case study that we performed in cooperation with an insurance company. The following section presents each step of the method in detail: Using examples from the case study, we show how to model the application structure as a dialog flow (Sect. 2.1), identify typical interaction sequences within the application (Sect. 2.2),
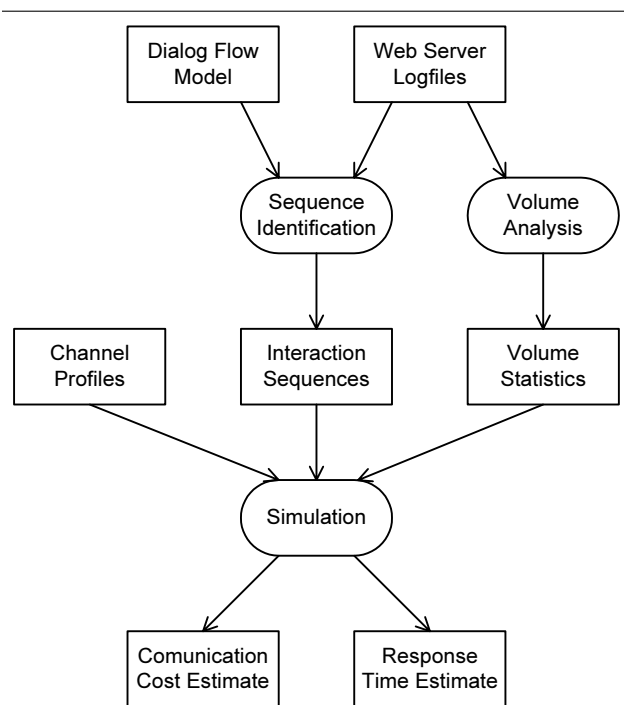
Figure 1. The PETTICOAT method

measure the time and data volume in the existing application (Sect. 2.3), specify the mobile channels' characteristics (Sect. 2.4), simulate the application's interaction sequences on different mobile channels and evaluate the usability and cost implications of the observed time and data volume (Sect. 2.5). After an overview of the related work in Sect. 3, Section 4 provides a short summary and outlines ongoing and future work in this area.

2. The PETTICOAT Method

The PETTICOAT method provides decision makers with indicators on the economical feasibility of mobile channel development. In a nutshell, it involves identifying interaction sequences in a dialog flow model of the existing application, measuring the time and data volume incurred in their execution (either by analyzing web server log files or observing real-time traffic), and then simulating how the same interaction sequences would perform when subjected to the frame conditions of a mobile channel. As a result of the simulation, we gain time and volume projections for the interaction sequences that allow us to estimate the cost incurred by working with the application on different mobile channels (Fig. 1).

The following subsections present these steps in more detail and illustrate them with excerpts from a case study we performed for an insurance company. In that project, we applied the PETTICOAT method to the prototype of a

new web-based offer management system in order to estimate the cost that will be incurred each month by insurance agents accessing the system over mobile networks such as GSM, GPRS and UMTS.

2.1. Modelling the Dialog Flow

As a basis for our analysis, we need a model of the application's complete dialog structure. We use the Dialog Flow Notation (DFN) [BG04] for this purpose. This graphical notation models an application's dialog flow as a directed graph of states that are connected by transitions. We call the transitions "events" and the states "dialog elements", distinguishing "masks" (web pages rendered on the client) and "actions" (business logic executed on the server). Events can carry parameters that transport business data such as form input.

By building such dialog graphs from masks, actions and events, the developer can specify all possible user interactions with the application. To increase the expressive power of the specification, dialog graphs can be encapsulated in "dialog modules" that can be reused in different contexts within the same application by nesting them into the dialog flow at arbitrary levels. This allows the developer to build complex dialog structures that closely mirror the users' mental model of the complex business processes supported by large-scale web applications.

As an example, Fig. 2 shows the dialog flow of the offer management system analyzed in the case study. Since
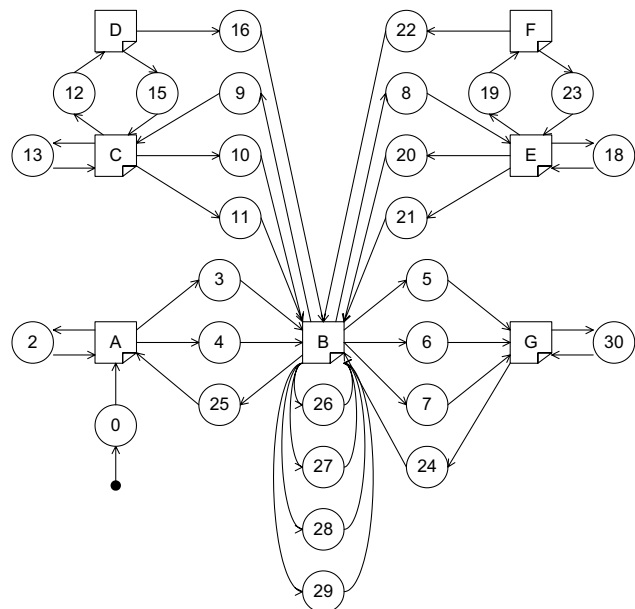

Figure 2. Dialog graph of the offer management system

Mask ID	Content
A	Search Transaction Form
B	Edit Transaction Form
C	Associate Agent Form
D	Create Agent Form
E	Associate Insurance Holder Form
F	Create Insurance Holder Form
G	Edit Offer Form

Action ID	Function
0	initialize system
2	search transactions
4	prep transaction for editing
6	prep offer for editing
...	...
26	expand transaction elements
27	collapse transaction elements
28	load documents
29	browse transaction elements
30	process offer modifications

Table 1. Masks and actions of the offer management system (excerpt)

we were looking at a rather simple prototype, the model does not make use of the DFN's dialog modularization capabilities and comprises only seven dialog masks connected through a number of actions that implement various business operations, an exemplary selection of which is shown in Table 1. Field staff users enter the application through the *initialize system* action *(0)*, which leads to the *Search Transaction Form* mask *(A)* where they can look up, create or edit transactions. Using the other masks and actions, transactions can be associated with insurance agents, insurance holders and policy offers.

In order to use these graphical specifications as input for the following steps, they can be automatically translated into the XML-based Dialog Flow Specification Language (DFSL) [BG04]. However, we do not show this straightforward conversion step here for the sake of brevity.

2.2. Identifying Interaction Sequences

The dialog graphs of an application specify all possible ways of interaction that the user interface allows. Since the same business process may be accomplished in a number of similar, but still different ways, there will typically be some more and some less frequently traversed paths through the dialog graph (called "interaction sequences" from now on). To arrive at a representative cost projection for the business processes performed with the application, we therefore need to analyze the actual interaction sequences that occur in the application. By identifying the sequences that the users tra-

verse most frequently, we can later weigh the cost they incurred accordingly.

In the case study, we identified and analyzed 15 interaction sequences (i.e. subsets of the whole dialog graph) of the offer management system. As an example, Fig. 3 shows the sequence for finding a transaction, browsing its elements and editing the associated offer. The events in this sequence are annotated with probabilities to reflect the different possibilities of executing this business process. Since a user's interaction steps are not isolated from each other, but depend on the history of his interactions, these probabilities are conditional: In the notation, we first note the probability of a user following this event, and then (after a vertical bar) note which action the user must have executed before as a condition for this probability. For example, from mask *A*, there is a 1.0 probability that the user will execute action *2* under the condition that he executed action *0* before, but a 0.5 probability that he will execute action *2* if he already executed that action before. In other words, if the user just entered the application, he will definitely use the search feature, but if he already searched for a transaction, there's only a 50% probability that he will use the search feature again. Rather, there's also a 50% probability that he will proceed to edit the transaction he found (denoted by the $0.5|2$ probability for the event leading to action *4*). For events without annotation, the implicit probability of traversal is 100%, regardless of the previously executed action.

One might argue that the events' probabilities may depend on a longer history than just the last executed action. Indeed, we are currently investigating the level of history that should be incorporated into this model in order to achieve sufficiently accurate approximations of the users' behavior. In our case study, the probabilities were estimated based on practical considerations. Alternatively, a more realistic probability model can be reached by evaluating user tracking information that is routinely collected in web server log files [PP99]. In complex web applications, however, the logged URLs may not always indicate unam-
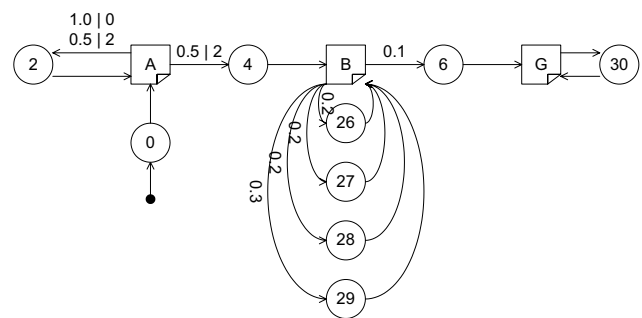

Figure 3. Interaction sequence for finding and editing an offer associated with a transaction

biguously which page was ultimately presented to the user. To increase the quality of the path identification, it may be necessary to log interaction data directly in the dialog control logic instead at the web server level. We're currently investigating ways of accomplishing this, ideally with non-invasive methods that do not require changes to the existing application logic.

While it is helpful to visualize the interaction sequences graphically in the conceptualization phase of the study, they need to be converted to a machine-readable format in order to be processed by the simulation tool. We use a variation of the DFSL for this purpose. The resulting sequence specification also contains estimates on how often each sequence will be executed by each user each month, which will be used towards the end of the simulation in order to calculate the approximate monthly cost of executing all sequences.

2.3. Measuring Data Volume and Time

As mentioned in the introduction, the two main factors influencing the cost of interaction with an application over a mobile channel are the time spent online and the data volume transmitted. To project these metrics for mobile channels, we measure them on the existing stationary channel and then input them into the simulation.

There are a few challenges in the details of this measurement process, however: Most importantly, for the volume measurement, we need to distinguish between static and dynamic content. While static content (such as images) always incurs the same volume (apart from caching effects, which can be accounted for in the simulation), dynamic content (such as search result pages) can produce a different volume for each request. To obtain accurate estimates, we need to deduce a probability distribution or an average

```
<WebElement>
  <ID>A</ID>
  <Desc>Search Transaction Form</Desc>
  <ElementType>text/html</ElementType>
  <Cache>0</Cache>
  <RequestUncomp>750</RequestUncomp>
  <RequestComp>500</RequestComp>
  <ResponseUncomp>9000</ResponseUncomp>
  <ResponseComp>3000</ResponseComp>
  <Inlines>
    <Inline>
      <WebElementID>100</WebElementID>
      <OffsetComp>1200</OffsetComp>
      <OffsetUncomp>5000</OffsetUncomp>
    </Inline>
  </Inlines>
</WebElement>
```

Figure 4. Web element specification for *Search Transaction Form* mask

value from the accumulated volume data. Also, web server log files only log the net volume of the content, but not any overhead introduced on lower levels of the protocol stack that nevertheless does count for billing purposes. This overhead can either be ascertained by observing the data flow directly on a sufficiently low protocol level instead of relying on server logfiles, or by factoring it into the simulation in close accordance with the respective protocol specifications.

In our case study, we used HttpWatch 3.2, a simple HTTP traffic listener [Sim05] to obtain the necessary data. The characteristics of each each web page, image etc. were described in an XML-based format where each of those "web elements" is represented by a `WebElement` tag that contains tags for its various attributes: Tags starting with `Request` or `Response`, for example, contain the data volume incurred for the request and the response of the web element in bytes, depending on whether the web server configuration allows HTTP compression or not. The `Inlines` tag contains references to web elements such as images included in a page. For each of these, we can specify the offset of their include point on the page (i.e. the number of bytes of the parent web element that need to be loaded before the inline web element is requested by the browser). As an example, the document excerpt in Fig. 4 shows the description of the *Search Transaction Form* mask in our case study.

To measure the time it takes to complete an interaction sequence, a number of contributing factors need to be considered. This total time a user spends online is the sum of user activity (e.g. filling in forms), upstream and downstream transmission time, channel latency and server processing time. To accurately distinguish all these contributing factors, we would need synchronized timing on both the server and the client. Fortunately, however, only the user and server activity matter for the subsequent simulation, since the observed transmission time and latency already depend on the stationary channel that we measured on. We can thus deduct them from the overall time during the simulation based on our knowledge of the stationary channel characteristics and volume transmitted. This way, we are left with the user and server activity time, to which we can add the newly calculated transmission time and latency based on the mobile channel's characteristics. These times are specified for each action, i.e. each transition between masks, in an XML-based format similar to the one shown in Fig. 4.

2.4. Defining Channel Characteristics

Besides the description of the application's interaction patterns, mask and action characteristics, we still need a detailed specification of the target (usually mobile) network
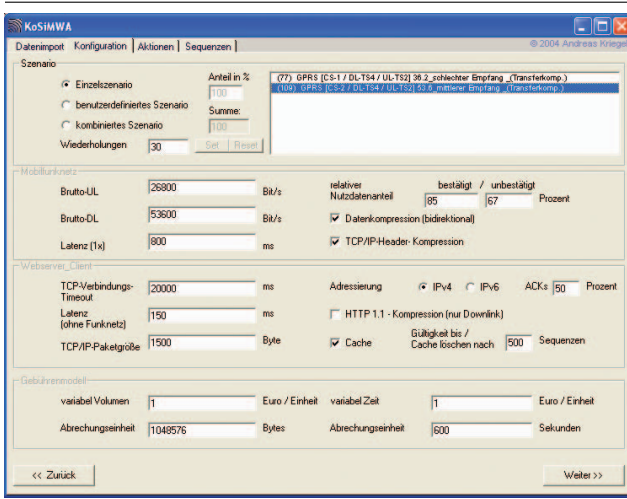
Figure 5. Specification of channel characteristics in the simulation tool

environment, since different mobile networks have different characteristics regarding bandwidth, latency, pricing etc. We define these characteristics in XML-based "channel profiles" for each network that shall be considered in the simulation. In our case study, we defined 16 channels, including different compression variants for GSM, HSCSD, GPRS and UMTS networks. The tool also considers effects of fluctuating signal strength. Each profile contains the gross uplink and downlink bandwidth in bits/s, as well as several attributes for packet and compression characteristics. Furthermore, the network provider's rates for volume-based and time-based billing are contained in the profile description. The rates in the case study were based on pricing plans of a German telecommunications provider. The channel profiles are stored in XML documents and can be edited by the user in the simulation tool, as illustrated by the definition of a GPRS 53.6 channel profile without data compression under the assumption of strong reception in Fig. 5.

2.5. Simulation of Interaction Sequences on Different Channels

In order to perform the simulation, our tool requires the XML documents produced in the previous steps as input, i.e. the application profile that contains the web elements and their volume data, the actions and their timing data, and the sequences with their probability and frequency data; and also the channel profiles containing the bandwidth, latency and pricing characteristics.

Using this input, the simulation tool works in three steps that will be described in detail in the following sections:

1. The simulator begins each interaction sequence at its entry point. Taking into account the branching probabilities, it then simulates the time it takes to load each mask in the sequence, considering the inline element offsets which incur latency and traffic that delay the completion of the mask. This step already yields insights into usability problems that may be caused by unacceptably high response times.

2. The results for each interaction step of a sequence are accumulated, taking the user's idle time between interactions into account.

3. The results for each interaction sequence are multiplied by its estimated frequency per user and month. Summing up the results finally yields the projected communication costs of the application per user and month.

2.5.1. Simulation of Interaction Steps. The simulation results for the response times of the actions (i.e. the transitions between the masks) in the interaction sequence shown in Fig. 3 on a selection of channels are given in Fig. 6. The diagram clearly shows that the use of a client-side cache (actions marked "with cache (w/c)") significantly reduces the response time in contrast to executing the same sequence without a cache (marked "no cache (n/c)"). However, it may still be relevant to investigate an application's response time without a cache since not all mobile platforms provide sufficient cache memory.

From Fig. 6, we can also determine that only the UMTS channel with enabled browser cache supports answering
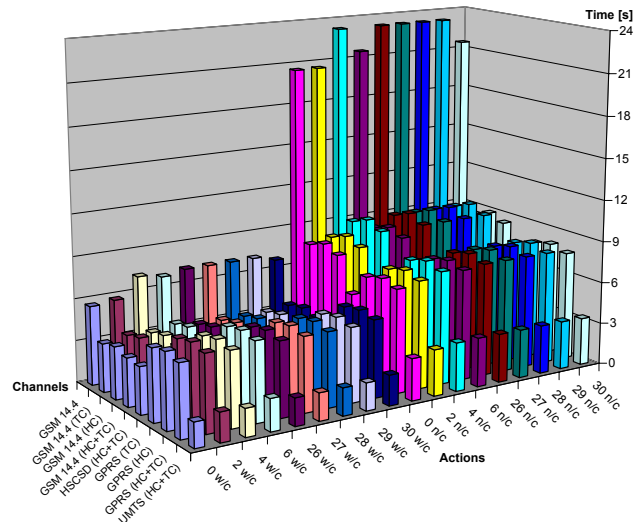

Figure 6. Simulation results for response times

times of less than three seconds for this application and thus provides adequate usability (if we follow Shneiderman's rule that response times for simple actions should not exceed one second, while four seconds are the maximum response time for standard actions [Shn02]).

Figure 6 also indicates that in our case study, the response times on the GPRS channel are longer than those on the GSM or HSCSD channel (using the identical compression and caching mechanism). This may come as a surprise, as GPRS may provide three to four times the bandwidth of a GSM 14.4 channel. On the other hand, GPRS has a network latency of about two seconds, so any request is delayed by about two seconds before the transmission of the requested data actually begins. By that time, the requested data would already have reached the recipient on the GSM channel. Only when the cache is deactivated, the GPRS channel can make up for the latency with its higher bandwidth.

Other, more complex timing constraints than the abovementioned three second usability rule are also conceivable. For example, we could define the constraint that mask n has to be loaded within t seconds after leaving mask m. The results gained in the simulation may then indicate which masks are responsible for failing the constraints and need to be optimized. If no redesign seems feasible, the application cannot be used on the simulated channel with the specified constraints. For example, in our case study, the results on the GPRS channel indicate that a redesign should particularly focus on embedding inline elements at the top of a mask, so that requests for these elements can be sent earlier by the browser and the network latency is mitigated by the initial page still being loaded.

2.5.2. Simulation of Interaction Sequences. In the second step, the simulation tool sums up the results for the individual steps in a sequence gained in the first step. It also adds the user's estimated idle time to simulate how long a user works with a mask on average before the next mask is requested. This way, the tool determines how long it typically takes to execute a whole interaction sequence on a channel (taking the different probabilities for the sequence variants into account), and how many bytes are transferred in the process. Using the providers' rates specified in the channel profiles, the tool can then calculate the cost of performing each interaction sequence on each channel.

In Table 2, an excerpt of the results gained for a selection of channels and sequences from our case study are given. For each of the available channels (GSM, HSCSD, GPRS and UMTS), four simulations were carried out using no compression, HTTP compression (by the web server), transfer compression (by the carrier), and both HTTP and transfer compression. Sequence 4 denotes the process of a user creating a new policy offer, which has to be associated with an existing insurance agent and a new insurance holder.

Sequence 11 represents the process of creating a new policy offer by copying an existing one. Finally, sequence 12 contains the results for finding and editing a policy offer, as shown in Fig. 3 and Fig. 6. For each channel/sequence combination, the table contains the time taken to execute the whole sequence, the total amount of kilobytes transferred in the process, and the cost incurred under a time- and volume-based pricing plan on the respective channel. Since volume-based billing is not available for GSM and HSCSD channels, the respective fields remain blank.

The results indicate that the use of data compression reduces the data volume to roughly a third of the uncompressed volume, resulting in lower transmission times. It is important to note, however, that when using transfer compression, the carrier will charge for the uncompressed data volume. HTTP compression thus seems to be the better choice for volume-based pricing plans, as only the reduced data volume is billed. This effect can be observed e.g. when comparing the results for scenario 11 (with and without cache) on the GPRS channel with transfer vs. HTTP compression. A volume-based plan also allows for more flexibility regarding idle times, since longer client-side activities before requesting the next mask are not billed. On the other hand, transfer compression seems to be the best choice for time-based plans, because it yields shortest transfer times resulting in lower charges. Combining both transfer and HTTP compression may combine their advantages, but due to a greater overhead and slightly longer execution time on the web server, this combination may not yield the lowest cost regarding time and/or volume.

2.5.3. Simulation of Monthly Usage. In the final simulation step, the tool uses the results gained so far to project the total cost that will be incurred when one user works with all interaction sequences in the application over the course of one month. This enables project managers to estimate the total communication costs that can be expected on all channels, and decide if the addition of a mobile channel will pay off.

For our case study, the final results indicated that a UMTS channel with combined transfer and HTTP compression and a volume-based pricing plan is the best option. This scenario would incur an estimated monthly cost of € 55.11 per user. A volume-based plan on a GPRS channel with transfer and HTTP compression would cost only € 54.94 per user and month, but exhibits worse usability due to the high network latency, as Fig. 6 illustrated. Since UMTS is currently not available all over the country, GPRS can still be recommended as a suitable backup solution with limited usability. The time-based plans for the HSCSD and GSM channel would result in monthly costs of € 298.35 and € 421.19 per user, respectively, with both using only transfer compression, since the combination of transfer and

Sequence	Cache?	Results	GSM 14.4	GSM 14.4 (Transfer Comp.)	GSM 14.4 (HTTP Comp.)	GSM 14.4 (HTTP Comp. + Transfer Comp.)	HSCSD (HTTP Comp. + Transfer Comp.)	GPRS (Transfer Comp.)	GPRS (HTTP Comp.)	GPRS (HTTP Comp. + Transfer Comp.)	UMTS (HTTP Comp. + Transfer Comp.)
4	w/c	Time [s]	81.4	63.9	66.2	64.0	62.9	75.2	76.1	74.6	52.0
		Volume [kB]	61	62	22	23	22	62	22	23	22
		Charge (t) [€]	0.41	0.34	0.35	0.34	0.28	0.13	0.13	0.12	0.09
		Charge (vol) [€]	–	–	–	–	–	0.06	0.02	0.02	0.02
4	n/c	Time [s]	206.2	105.5	108.9	105.3	76.5	89.2	90.6	88.6	58.7
		Volume [kB]	292	292	109	109	109	293	109	109	109
		Charge (t) [€]	1.03	0.56	0.57	0.55	0.34	0.15	0.15	0.15	0.10
		Charge (vol) [€]	–	–	–	–	–	0.29	0.11	0.11	0.11
11	w/c	Time [s]	185.4	121.1	130.9	144.3	129.0	156.2	160.5	154.4	110.2
		Volume [kB]	143	126	40	45	41	137	42	41	43
		Charge (t) [€]	0.93	0.64	0.69	0.76	0.58	0.26	0.27	0.26	0.18
		Charge (vol) [€]	–	–	–	–	–	0.13	0.04	0.04	0.04
11	n/c	Time [s]	420.8	219.0	230.2	212.2	165.3	191.4	203.4	199.5	124.7
		Volume [kB]	580	577	222	209	220	566	221	222	212
		Charge (t) [€]	2.10	1.16	1.21	1.12	0.74	0.32	0.34	0.33	0.21
		Charge (vol) [€]	–	–	–	–	–	0.55	0.22	0.22	0.21
12	w/c	Time [s]	210.1	157.3	154.6	162.5	149.1	191.0	184.7	182.9	129.9
		Volume [kB]	150	150	43	47	43	152	45	45	45
		Charge (t) [€]	1.05	0.83	0.81	0.86	0.66	0.32	0.31	0.31	0.22
		Charge (vol) [€]	–	–	–	–	–	0.15	0.04	0.04	0.04
12	n/c	Time [s]	467.5	250.9	257.0	248.5	193.8	222.4	223.8	217.3	150.8
		Volume [kB]	621	622	231	230	236	620	226	223	230
		Charge (t) [€]	2.34	1.32	1.35	1.30	0.87	0.37	0.37	0.36	0.25
		Charge (vol) [€]	–	–	–	–	–	0.61	0.22	0.22	0.23
...											

Table 2. Simulation results for performing interaction sequences on different channels (excerpt)

HTTP compression would be even more expensive in total.

3. Related Work

In [DVDR01], it is shown how frequent and thus critical user paths can be identified in e-commerce applications. The authors provide a model of the user behavior in form of session graphs and conduct analyses regarding the most frequently used user paths as well as critical edge sequences. This technique could be quite useful for our approach, because the identification of the most frequently used subset of all possible user paths in the application model is needed.

Furthermore, there are lots of approaches for web log analysis aimed at classifying user paths, e.g. [Spi00], [BBR01], [KAB+04], [HC02], [CPP00], [GSdC00]. Especially the identification of long sequences [PP99] seems to be an important topic for the PETTICOAT concept. The identification of actually chosen user paths vs. all possible user paths in the application model is needed in order to obtain meaningful results from the following simulation. In this context, the work of [MCWZ01] is of specific interest. They present a notion for a cluster-based on-line monitoring system for web traffic. The target-oriented analysis of web traffic is a task to be solved within the PETTICOAT approach.

As PETTICOAT particularly addresses the analysis of dynamic web applications instead of static web pages, the analysis of web traffic is even more difficult. This problem is addressed e.g. in [BS00], which deals with dynamic web content generation and website analysis.

Other approaches to improving the performance of web-based applications have focused on using thin clients to transmit just the image of the application (see e.g. [LNB+04]). The findings of this work are of relevance for the deduction of consequences (application design, bandwidth restriction) based on the simulation results. In this context, [BRVX04] and [KW00] report interesting results from an analysis of large websites regarding performance, cache and cookie issues. These results could

be used for the creation of a package of measures in order to modify the analyzed website regarding performance issues in the mobile environment.

4. Conclusions and Future Work

In this paper, we have shown a method for assessing the usability implications and communication costs of adding mobile channels to an existing web-based application. As illustrated by the case study, the results of the simulation indicate if an existing application can be accessed efficiently on certain mobile channels, and provide clues on how the application may have to be optimized for lower response times. The simulation also provides an estimate of the cost of using the application on various mobile channels, which is a valuable factor in deciding if the introduction of a mobile channel will pay off for an organization in the future.

In our ongoing work, we are currently focusing on automated analysis of web applications to simplify the initial steps of the PETTICOAT method. This includes deriving the dialog flow model and the probabilities and frequencies of typical interaction sequences from the data contained in web server log files, rather than modeling them manually. Further research will comprise refinements of the probabilistic model for the interaction sequences and a more detailed specification of mobile channel characteristics and billing schemes in order to increase the accuracy of the estimates, and thus the quality experienced by users when accessing web-based applications through mobile channels.

5. Acknowledgements

References

[BBR01]   Pavel Berkhin, Jonathan D. Beche, and Dee Jay Randall. Interactive path analysis of web site traffic. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 414–419. ACM Press, 2001.

[BG04]   Matthias Book and Volker Gruhn. Modeling web-based dialog flows for automatic dialog control. In *19th IEEE International Conference on Automated Software Engineering (ASE 2004)*, pages 100–109. IEEE Computer Society Press, 2004.

[BRVX04] L. Bent, M. Rabinovich, G. M. Voelker, and Z. Xiao. Characterization of a large web site population with implications for content delivery. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 522–533. ACM Press, 2004.

[BS00]   Bettina Berendt and Myra Spiliopoulou. Analysis of navigation behaviour in web sites integrating multiple information systems. *The VLDB Journal*, 9(1):56–75, 2000.

[CPP00]   Ed H. Chi, Peter Pirolli, and James Pitkow. The scent of a site: a system for analyzing and predicting information scent, usage, and usability of a web site. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 161–168. ACM Press, 2000.

[DVDR01] Kaushik Dutta, Debra VanderMeer, Anindya Datta, and Krithi Ramamritham. Discovering critical edge sequences in e-commerce catalogs. In *EC '01: Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 65–74. ACM Press, 2001.

[GSdC00] Mark Gillenson, Daniel L. Sherrell, and Lei da Chen. A taxonomy of web site traversal patterns and structures. *Commun. AIS*, 3(4es):5, 2000.

[HC02]   Jeffrey Heer and Ed H. Chi. Separating the swarm: categorization methods for user sessions on the web. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 243–250. ACM Press, 2002.

[KAB+04] Dong-Ho Kim, Vijayalakshmi Atluri, Michael Bieber, Nabil Adam, and Yelena Yesha. A clickstream-based collaborative filtering personalization model: towards a better performance. In *WIDM '04: Proceedings of the 6th annual ACM international workshop on Web information and data management*, pages 88–95. ACM Press, 2004.

[KW00]   Balachander Krishnamurthy and Craig E. Wills. Analyzing factors that influence end-to-end web performance. In *Proceedings of the 9th international World Wide Web conference on Computer networks : the international journal of computer and telecommunications netowrking*, pages 17–32. North-Holland Publishing Co., 2000.

[LNB+04] Albert M. Lai, Jason Nieh, Bhagyashree Bohra, Vijayarka Nandikonda, Abhishek P. Surana, and Suchita Varshneya. Improving web browsing performance on wireless pdas using thin-client computing. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 143–154. ACM Press, 2004.

[MCWZ01] Yun Mao, Kang Chen, Dongsheng Wang, and Weimin Zheng. Cluster-based online monitoring system of web traffic. In *WIDM '01: Proceedings of the 3rd international workshop on Web information and data management*, pages 47–53. ACM Press, 2001.

[PP99]   James Pitkow and Peter Pirolli. Mining longest repeating subsequences to predict world wide web surfing. In *Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems*, 1999.

[Shn02]   Ben Shneiderman. *User Interface Design*. mitp-Verlag, 2002.

[Sim05]   Simtec Limited. HttpWatch 3.2. http://www.simtec.ltd.uk, 2005.

[Spi00]   Myra Spiliopoulou. Web usage mining for web site evaluation. *Commun. ACM*, 43(8):127–134, 2000.