IST FP7 231507

D5.2 Development of new user centered evaluation measures

File name	PuppyIR-D5.2-Development-of-new-user-centered-evaluation-measures
Author(s)	Desmond Elliott (UGLW) Leif Azzopardi (UGLW) Tamara Polajnar (UGLW) Richard Glassey (UGLW)
Work package/task	WP5/Task 5.2
Document status	Final
Version	1.0
Contractual delivery date	M28
Confidentiality	Public
Keywords	User Centered Evaluation
Abstract	This report describes a suite of new user-centered evaluation measures based on a stream-based view of the interaction process. The basis of the stream-based view is outlined before the meaures are presented and demonstrated in both a simulation and observational context.

Table of Contents

Executive Summary

This report provides the conceptual framework for measuring the performance of PuppyIR systems. The applications developed using the PuppyIR framework are primarily designed for children. Children have a different approach to information-seeking than adults. They resort to browsing more often than query-based searching due to difficulties with typing and query formulation. The traditional list-based evaluation is not appropriate for non-linear interactions, or indeed for application evaluation. Therefore in this report we outline a suite of measures more geared towards evaluating a variety of interactions.

The aim of an information-seeking support system is to assist the user in accessing relevant information effectively and efficiently. It is well known that system performance, in terms of finding relevant information, is heavily dependent upon the user interactions. A pragmatic evaluation question that arises is: what is the effectiveness experienced by the user during the usage of the system? This question is especially relevant when considering how children use these types of systems, because it has been shown that their browsing behaviours are more erratic than adults. To be able to answer this question, we represent the usage of a system by the stream of documents the user encounters while interacting with the system. This representation enables us to monitor and track the performance over time. By taking a stream-based view of the interaction process, instead of a ranked-list view, the evaluation can be performed on any type of system.

In the first three sections we outline the motivation and the measures, and in section four we describe how these could be demonstrated in a simulated environment and for an implemented prototype.

1 Introduction

The primary goal of an information-seeking support system, such as those being developed for the PuppyIR project, is to provide efficient and effective access to relevant information. An information retrieval system is one example application of these types of systems, in which users typically submit text queries to search for information. The output of interacting with this application is usually a ranked list of documents, which are subsequently judged by the user for relevance. An alternative application is an information filtering system, where users specify their long-term interests and the application filters documents from the stream of content published on a daily basis. Same data can be accessed using either of the systems. The filtering system provides timely access of relevant data as it is being published, while the retrieval system enables search for explicitly specified topics in the previously published documents. Due to the difference in the definition of the information need, it is difficult to compare the systems directly.

The effectiveness of information retrieval systems has traditionally been studied in a simulated environment without active users. This evaluation framework was inspired by the Cranfield experiments conducted as part of the S.M.A.R.T.project [Cleverdon et al. 1966], where system effectiveness was determined using relevance judgements. This framework evolved into the Text REtrieval Conference [Voorhees and Harman 2005], where the effectiveness of these systems has been studied on a large-scale.

An open challenge is the evaluation of information-seeking support systems in an interaction context. This report proposes that a generalised view of the interaction process can address some of the difficulties in these types of evaluation. Interactions are represented as a stream of events, requiring a novel approach for measuring algorithm effectiveness. A suite of measures around this view is proposed.

A further challenge is how to perform effectiveness evaluations when the users are children [Bilal and Kirby 2003]. The search and browsing behaviour of children has been shown to differ from that of adults. Children have difficulty using query-based interfaces: there are cognitive barriers in query formulation (not necessarily unique to children); and a lack of typing dexterity results in children having difficulty observing the screen and keyboard at the same time. This leads to typographical errors and failure to notice events such as suggested queries. Secondly, children have a lower attention span and experience difficulty in judging the relevance of a document, which causes them to browse more erratically than adults. This leads to frequent revisiting of previous pages or ‘looping’, as well as spending less time reading a page.

The remainder of this report is structured as follows.

- Section 2 presents and argues for the stream-based view of applications. The limitations of the ranked-list approach are presented and the details of the stream-based approach are described.
- Section 3 outlines the evaluation measures which can be used under a stream-based view of the iterative process with an application.
- Section 4 demonstrates the stream-based view and evaluation measures in a simulated environment using both an information retrieval and information filtering application.
- Section 5 demonstrates these evaluation measures in an interaction environment using an information filtering application. This application has been developed as part of the PuppyIR project and is currently being used by several partners to prototype their deliverables.

2 Stream-based Analysis

The differences between information-seeking support systems make it challenging to evaluate the effectiveness of each application. The evaluation difficulties arise because of three main reasons:

1. Interaction is difficult to synthesise and replicate, which makes the creation of a TREC-like test collection for interactive evaluations unfeasible [Voorhees 2008];

2. Standard evaluation focuses on:
   a) the system, method or model, and not the application
   b) the topic or task, and not the usage over time and the user experience, and;

3. Measurements are typically based on a ranked list with a prescribed order of assessment and this does not generalise well to interactive scenarios [Belkin 2008].

The evaluation of different interactive systems requires an evaluation paradigm that differs from the Cranfield convention. One that is more general, at a higher level, and focuses on usage. It needs to be general in the sense that any type of application can be evaluated, i.e. at the application level, to incorporate interaction with the interface within the evaluation. It also needs to be focused on usage, because how an application is used determines how much relevant information is accessed and thus how effective it is.

2.1 The Ranked-list View

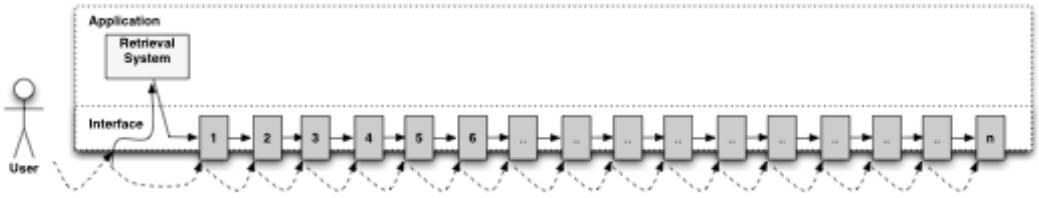

Figure 1: Sequence diagram of a standard information retrieval application.

In the standard model of the information retrieval process, a user submits a query to the system for a given topic and the system responds by presenting the user with a ranked list of documents. The user then assesses, in turn, each document in the ranked list. Figure 1 depicts the standard IR process. This ranked list view makes several assumptions, regarding the interaction and usage of the application:

- the process is initiated by a query for a given topic - the information need is fixed;
- the documents are presented in a ranked list;
- the user inspects documents, sequentially and in order; and
- the user inspects the entire ranking up to a cut off point n.

This abstraction of the process has been the basis of much of the evaluation performed for information retrieval systems. Consequently, most evaluations use measurements assuming a ranked list. However, an application, and the way it is used, may not necessarily produce a ranked list of documents; nor may the user inspect the documents in a linear fashion. Various strategies may be employed by the user during the process, for example: inspecting document clusters, using a find similar feature, browsing links or facets. This assumed user behaviour is seldom the case in practice and so this view of the process does not generalize well to non-standard applications or non-deterministic usage.

One direction that has been taken to evaluate non-standard interaction is to transform the output of the interaction into a ranked list enabling the comparison against standard ranked based methods. For example, in [Urban et al. 2003], an ostensive browser application recommends similar documents, given the previous documents viewed. The trail of documents the user visits is used to form a ranking, which could be compared to a standard retrieval method's ranked list. In [Leuski 2001, Lin and Smucker 2008, Smucker and Allan 2007, White et al. 2005] rankings are formed from the interaction with the envisioned application. While transforming the sequence of documents encountered into a ranked list is appropriate for some applications, it is not possible for all applications. For example, a filtering application recommends documents, and so the notion that it is evaluated as a ranked list is not appropriate. Nor is it appropriate in an exploratory search application where the query is ill defined, and the information need is dynamic [Belkin et al. 1982]. While reverting to a ranked-list enables comparison with standard retrieval models, it does not consider evaluation at the application level, where it is important to observe the effectiveness that the user experiences during the usage of the application. We argue that adopting a stream-based view provides a general way to represent the usage of an application, such that filtering and retrieval applications are interchangeable. In particular, it enables the measurement of the effectiveness as experienced by the users throughout their interactions with the application.

2.2 The Stream-based View

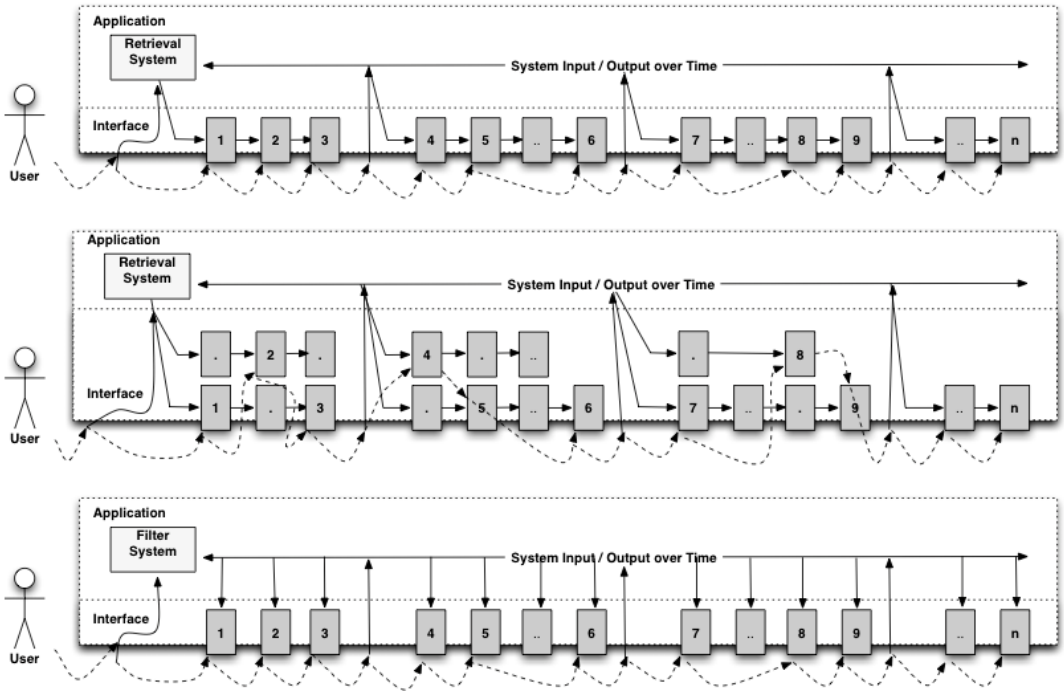

Figure 2: Sequence diagrams for interactive information retrieval, cluster-based retrieval, and an information filtering application (top, middle, bottom). The dotted line denotes the user interacting with the application.

A stream-based view of the sequence of interactions with a system is proposed to address the problems of the ranked-list view, as advocated by [Bookstein 1983]. In the stream-based view, the interaction between the user and the application produces a stream of documents, which are assessed during the usage of the application. The sequence of interactions determines the user's perception of the application's performance, which in turn defines their experience [Norman

1988]. From the user's point of view, it is the stream of documents they interact with that largely determines their experience of the application, because their goal is to access useful information. Consequently, the application needs to deliver a sufficient amount of useful information for the user to have a satisfactory experience. From an application provider's perspective, it is important that the usage of the application results in a good experience; a poor user experience may lead to disengagement or even abandonment of the system.

The origins of the stream-based view of interactions with an application, the user examines a sequence of documents and the system receives feedback and adjusts the documents presented to the user. Depending on the feedback, the system presents different documents and the interactions create a sequence of accessed documents. Bookstein argues that it is this sequence of documents that should be evaluated. In [Azzopardi 2007], this view is generalised to any information-seeking support system, such that the usage of the system results in a stream of documents presented and assessed by the user. For any given system, the interaction can be characterised as follows:

1. The user performs an action given the system interface.

2. The application engages the underlying system to produce a response, which is then presented to the user via the interface.

3. The user assesses the response and engages with the presented documents, then the user performs a subsequent action, and so on.

The order in which the user engages with the presented documents defines a stream of documents. This stream-based view does not make any assumptions about the user actions (it does not have to be a query), or how the documents are presented to the user (it does not have to be a ranked list). Without the standard assumptions, it is possible to generalise the stream-based view to consider any type of application. Some example applications, that are all represented using this view, are shown in Figure 2. This sequence diagram shows how the stream of documents (denoted by 1,., n) is built over the course of interaction with the interface.

The main difference between this stream-based view and the ranked-list view is that the former is temporal and usage specific. The stream of documents encountered by the user depends upon how the application was used. It is this stream that forms the basis of the evaluation of the effectiveness that the user experiences. While the stream-based view imposes few restrictions on describing the interaction process, it does require other assumptions:

1. Only one document can be interacted with at a time, and

2. Each document is independently judged, each time that it is accessed.

The first assumption is that the user can only access one document at any particular point in time. There are, of course, instances when parallel streams of documents are encountered during the usage of a system. This may be the case when dealing with images, or comparing multiple documents in multiple windows. However, in general, only one document is examined at any one point in time so it is reasonable to engage this simplifying assumption in order to develop usage based measures for streams.

Regarding the second assumption, in the ranked-list view it is assumed that each document is judged (independently) with respect to a topic. It has, however, been widely acknowledged that through interaction, the information need changes as the user's state of knowledge changes. Under the stream-based view, it is assumed that each document is judged with respect to the user's current information need and state. This is an important point because it means that when the same document appears in the stream, it may attract a different relevance judgement by the user. These assumptions mean that it is the usage of the application that is evaluated and not

whether the system is able to achieve total recall. Thus, usage based effectiveness measures will be predominately precision oriented.

3 Usage-based Evaluation Measures

In this section, we describe a suite of measures for tracking and monitoring the performance resulting from the usage of an information-seeking support system under the stream-based view. These measures are designed for use in either a simulation or observational study.

First, we begin with the necessary definitions and notation before outlining the new evaluation measures that provide an estimate of the usage performance at a particular point in the stream, or for a particular time period within the stream. Note that since an application's performance is monitored in the context of usage, it is not possible to develop recall-based measures. This would require a post-hoc assessment of all documents. Such a task is problematic due to changes in a user's information need over time and context [Harter 1992]. We introduce a novel measure of the performance, called relevance frequency, which characterizes the rate at which relevant information is encountered during the usage of an IR application.

3.1 Preliminaries: Notation and Definition

Before we outline the notation, it is necessary to clarify the definitions of streams and sub-streams. A stream is a sequence of objects ordered temporally. A sub-stream is a sub-sequence derived from the stream, where the order of the objects is preserved. For the purposes of measurement the stream is decomposed into sub-streams. This can be performed, logically, conceptually or practically depending on the specific unit of interest, for example by topic, session, hour, or day.

To formalize the measurement of streams, we first introduce some notation. Let us denote a stream, which consists of a sequence of documents as $s = (d_1,d_2,...,d_N)$ with length N. For each document d_i we assume that there is an associated judgement r_i assigned to it forming a corresponding sequence $r = (d_1,d_2,...,d_N)$. A stream s can be decomposed into sub-streams, such that $s_{ij} = (d_i,d_{i+1},...,d_j)$. We shall use stream and sub-stream interchangeably.

For the purpose of introducing the set of measures we focus on a dichotomous decision based on document relevance, where $r_i = \{0, 1\}$. However, the value of r_i could also be a rating, grade or a continuous measurement. This judgement represents whether the document is relevant or useful to the user at that time point in the stream.

For a given stream s with the corresponding sequence of judgements r, it is possible to estimate the precision of the stream by treating the stream as a set and determining the proportion of relevant documents within the stream:

$$Prec(s) = \frac{1}{N}\sum_{i=1}^{N} r_i$$

where

$$r_i = \begin{cases} 1, & \text{if } d_i \text{ is relevant} \\ 0, & \text{otherwise} \end{cases}$$

Given a series of sub-streams ordered by time, it is then possible to obtain a series of precision measurements across the entire stream. While the individual sub-streams are treated like sets,

the order of the measurements determines the usage performance experienced over the course of interaction.

3.2 Precision Stream Measures

Depending on the type of application and the focus of the evaluation the stream is decomposed into sub-streams, accordingly. This defines the unit of measurement. While there are many possible ways to decompose the stream, here we only consider a few possible variations (and leave other variations for future work):

- Precision Blocks: the stream is decomposed as contiguous sub-streams s_{ij} of equal length N.
- Precision Windows: A stream is decomposed into overlapping sequences of equal size N. In other words a window is moved across the stream to create sub-streams (i.e. $s_{i,j}$, $s_{i+1,j+1}$, $s_{i+2, j+2}$, ...)
- Precision Day/Week/Month: A stream is decomposed into contiguous sub-streams according to a time unit such as hour, day, week, month, etc, resulting in sub-streams of different length (as the number of documents accessed in a given time frame may vary).
- Precision Session/Topic: A stream is divided into sequences of variable length determined by a (user) session or topic, i.e. streams can be decomposed such that they are ranked lists, where precision stream measures can be applied, as well as the numerous other rank-based evaluation measures (if desired).

Regardless of the decomposition, these measures will provide an indication of the application's performance over time, enabling the monitoring of performance. The main distinction between the Block and Window measures, and the other measures, is that they do not consider the time between document interactions, but simply the order; while the other measures consider the period of time in which the usage took place. As we shall see (in the next section) both provide interesting ways in which to track, monitor and analyze the usage performance.

- Cumulative Average Precision: As the precision measures provide point estimates of performance for the sub-streams, it is of interest to summarize the usage performance experienced over these sub-streams (i.e. a temporal average). The cumulative (marco) averaged precision (CAP) can be obtained by averaging over the measurements taken on a stream s decomposed into M sub-streams s_j, as follows:

$$CAP(S_M) = \frac{\sum_{j=1}^{M} Prec(s_j)}{M}$$

this represents the cumulative distribution of precision in the stream up to and including sub-stream s_M . We refer to this as a macro-average since the average is taken based on the precision of the sub-streams, and differs from the micro- average which be estimated at the document level. Measurements of this nature indicate how the application's usage performance convergences over time/usage. Other statistics, such as the standard deviation and standard error for the stream precision measurements can also be calculated.

3.3 Relevance Frequency

In the previous subsection, we defined a suite of precision-based measures, however, this only provides one possible way to evaluate streams. In this subsection, we propose a novel stream

based measure, which conveys a different view of the usage performance. Intuitively, the number of documents a user must examine before encountering a relevant document will impact upon their user experience. If they encounter many non-relevant documents successively this will detract from the user experience. And, many long periods of non-relevance before encountering relevant information is likely to lead to a negative user experience. The Relevance Frequency measure aims to quantify the rate at which relevant documents are encountered during the stream.

- RFreq: Given a stream s, decompose s into sub-streams by partitioning the stream whenever a relevant document occurs. The length x of a sub-stream denotes how many documents are examined to find a relevant document. So the Relevance Frequency for a distance of x is the count of the number of sub-streams that are of length x, the RFreq(x). For example, if the judgements on a stream yields:

  { | R | R | N, R | N, N, R | N, N, N, R | }

where the "|" indicates the sub-streams. Then, RFreq(1) = 2 because there are two sub-streams of length one, RFreq(2) = 1, RFreq(3) = 1, RFreq(4) = 1, and RFreq(> 5) = 0. By plotting the RFreq(x) values the distribution of encountering relevant documents can be visualized.

- Points of Failure (pof): If it is crucial that the application delivers relevant information at least every y documents, then it is easy to compute the number of points in the stream where this criteria is not satisfied, i.e. the number of points of failure is equal to:

$$pof(x > y) = \sum_{x>y} RFreq(x)$$

Obviously, a stream which contains all relevant documents will result in a RFreq(x = 1) = n, where n is the length of the stream, and RFreq(x > 1) = 0. Whereas a stream which contains only non-relevant documents will result in an RFreq(x = i) = 0 for all i ≤ n. This is because no relevant document occurs in the stream, so no sub-streams, which are terminated by a relevant document, exist within the steam.

- EFreq: To summarize the distribution of the Relevance Frequencies, the maximum likelihood estimate of the distribution can be taken to obtain the Expected Relevance Frequency as follows:

$$E[RFreq] = \frac{\sum_x x \times RFreq(x)}{\sum_x RFreq(x)}$$

where E[RFreq] denotes the expected rate of relevant documents in the stream. If E[RFreq] = 10 the user could expect to encounter nine non-relevant documents before encountering a relevant document at the tenth position, on average. It should be noted that the Expected Relevance Frequency is the stream-based analogy to Cooper's Expected Search Length [Cooper 1968], which is computed given a ranked list.

An empirical demonstration of these measures in a series of simulated scenarios can be found in [Azzopardi 2009].

4 Applying Stream-based Measures in PuppyIR

Within this project, the main concern is to evaluate the quality of applications based on the implicit feedback from the children. Children have a different approach to using the Internet, a more browsing-based approach. Many observational studies are available on the subject of the child-computer interaction [Bilal and Kirby 2003]; however, there are not many that evaluate the level of *engagement* children have with particular applications, and the *precision* of these applications. The novel contributions within PuppyIR can concentrate on measuring these two aspects:

A. Engagement is evaluated by observing the proportion of presented items that draw the child's attention, which can be judged, for example, by whether an item was clicked.
B. Precision is measured by how many of the clicked items were relevant, where relevance is assumed if some explicit action is pursued, such as following the link to the full story, or spending *x* amount of time on a specific item.

This section shows an example evaluation study using a prototype filtering application, which is described in more detail in D4.3 Report on Implementation and Documentation and in [Elliott et al. 2010, Glassey et al. 2010].

4.1 Example Application

In FiFi, latest news feeds for children are presented in order of time. The feeds can also be followed by topic, the list of which can be edited by adding new topics, or by closing topics. The interest in the topic can thus be explicit, or implicit (the number of times a child chooses a particular topic, or the stories associated with this topics). Engagement with stories can be assessed by the number of stories a child previews, while precision can be assessed with the

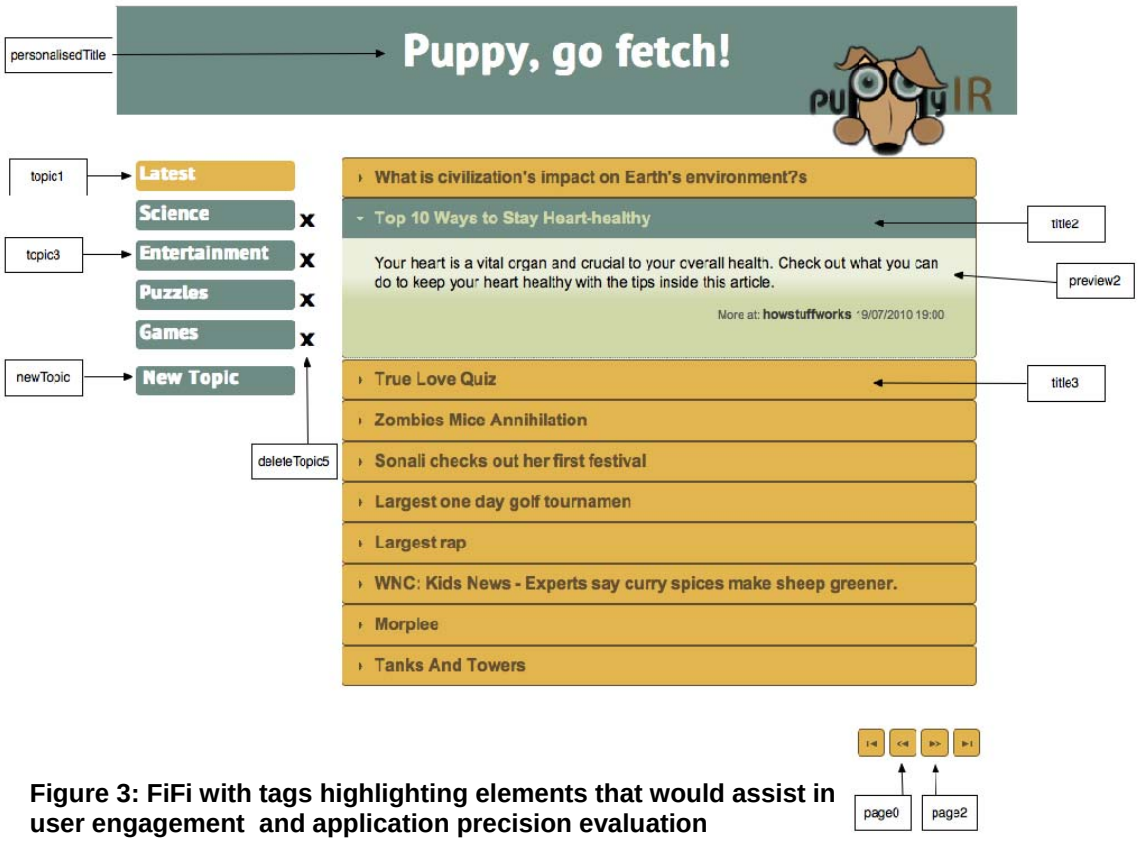

Figure 3: FiFi with tags highlighting elements that would assist in user engagement and application precision evaluation

number of stories that are followed up. A relevance window can be a page, which in this case is 10 stories long, but could be altered to be shorter.

Capturing of the user interactions can be achieved by video recording the user and manually annotating the application usage, or by implicitly logging the interactions through the browser. The latter approach might be more acceptable for parents and children, as it is less intrusive than collecting video. (We don't focus on the development of query log analysis tools; this is described in D5.2 Development of Query Log Analysis Tools.)

Though logging software can be custom-written for each application, there is an available package called UsaProxy.

Figure 4: UsaProxy system architecture

UsaProxy [Atterer et al. 2006] captures interactions by sitting in between the client and the application server as a transparent proxy server, as shown in Figure 3. When a client accesses the application, UsaProxy injects a Javascript library, which has been designed to track all interactions with the application. The interactions that can be tracked for the FiFi application are:

- Mouse movements;
- Mouse clicks; and
- Key presses.

UsaProxy reports detailed information on many user interactions. A sample of the interaction data with the FiFi application can be seen in the text box below.
The IP address of each user is reported, alongside the date and time of each tracked interaction. The URL of the origination of the interaction is also logged alongside the type of interaction. The sample data shows the user connecting to the application and clicking on the interface element relating to *topic1* – the Latest topic. The user then clicks on the title of the second item – *title2*, and then clicks on *preview2*, which takes them to the actual content, as shown in the last line of the sample.

Each application needs to be prepared on a case-by-case basis. This is unavoidable because of differences between applications and platforms, for example an application interface developed for a surface computing device compared with a Web-based application.

The FiFi interface can be seen in Figure 3, which includes notes on the different types of markings used to support the use of these measures. The HTML has been annotated to facilitate accurate interaction tracking. The HTML elements of interest to usage-based tracking are the topics, the item titles, and the item preview areas. Each topic has a unique identifier, which are directly related to the database representation of each topic. Each item title identifier is directly related to the document reference in the document index, along with the item previews.

141.84.8.77 2005-10-25,11:5:57 http://pooley:8080/fifi/ serverdata 12
141.84.8.77 2005-10-25,11:5:58 http://pooley:8080/fifi/ load width=1280;height=867
141.84.8.77 2005-10-25,11:6:2 http://pooley:8080/fifi/ mousemove x=672;y=7
141.84.8.77 2005-10-25,11:6:6 http://pooley:8080/fifi/ click x=815;y=231 target=id:topic1
141.84.8.77 2005-10-25,11:6:37 http://pooley:8080/fifi/ mousemove x=849;y=352
141.84.8.77 2005-10-25,11:6:37 http://pooley:8080/fifi/ mousedown x=161;y=229 target=id:title2
141.84.8.77 2005-10-25,11:6:40 http://pooley:8080/fifi/ mousemove x=148;y=138
141.84.8.77 2005-10-25,11:6:50 http://pooley:8080/fifi/ click x=26;y=507 target=id:preview2
141.84.8.77 2005-10-25,11:47:45 http://health.howstuffworks.com/diseases-conditions/cardiovascular/heart/10-ways-to-stay-heart-healthy.html scrolledTo y=399

Figure 5: Example UsaProxy Log

5 Conclusion

In this report, we have formalized an alternative approach to the evaluation of interactive information-seeking support applications. The stream-based view that forms the basis of this approach represents the usage of any application through a stream of documents. Given the goal of an application, the proposed usage-based effectiveness measures provide a novel way of monitoring and modelling the performance experienced by a user while interacting with the application.

There are two practical considerations that need to be addressed in order to monitor the usage performance of an application:

- how to build up the stream of documents (and how to deal with un-assessed but presented documents, etc), and;
- obtaining judgements on the utility/relevance of each document encountered in the stream (preferably, implicitly and unobtrusively).

We have assumed that every document in the stream has a corresponding judgement (i.e. the completeness assumption), however this does not mean that the stream cannot contain unassessed documents. Considering unassessed documents would require further measures to be developed. These could reflect other aspects of the user experience, such as user engagement, which poses a greater challenge. There have been many advances made towards developing mechanism to infer the relevance of documents implicitly [Kelly and Teevan 2003, Kelly 2004, White and Kelly 2006]. As these mechanisms improve and the implicit judgements become more reliable, the quality and accuracy of the usage performance measures will also improve making this form of evaluation feasible and more reliable.

References

[Cleverdon et al. 1966] C. W. Cleverdon, J. Mills, M. Keen, Aslib Cranfield research project - Factors determining the performance of indexing systems; Volume 1, Part 1, Design; Part 2, Appendices; Volume 2, Test results.

[Voorhees and Harman 2005] E. Voorhees and D. Harman. TREC: Experiment and Evaluation in Information Retrieval. MIT Press, 2005.

[Bilal and Kirby 2003] D. Bilal and J. Kirby. Differences and similarities in information seeking: children and adults as web users. Information Processing & Management, 38(5):649–670, 2002.

[Voorhees 2008] E. Voorhees. On test collections for adaptive information retrieval. IPM, 44(6):1879–1885, 2008.

[Belkin 2008] N. J. Belkin. Some(what) grand challenges for information retrieval. SIGIR Forum, 42(1):47–54, 2008.

[Urban et al. 2003] J. Urban, J. M. Jose, and C. J. van Rijsbergen. An adaptive approach towards content-based image retrieval. In Proc. of CBMI'03, pages 119–126, 2003.

[Leuski 2001] A. Leuski. Evaluating document clustering for IIR. In Proc. of CIKM '01, pages 33–40, 2001.

[Lin and Smucker 2008] J. Lin and M. D. Smucker. How do users find things with pubmed?: towards automatic utility evaluation with user simulations. In Proc. of SIGIR '08, pages 19–26, 2008.

[Smucker and Allan 2007] M. D. Smucker and J. Allan. Using similarity links as shortcuts to relevant web pages. In Proc. of SIGIR '07, pages 863–864, 2007.

[White et al. 2005] R. W. White, I. Ruthven, J. M. Jose, and C. J. van Rijsbergen. Evaluating implicit feedback models using searcher simulations. ACM TOIS, 23(3):325–361, 2005.

[Belkin et al. 1982] N. J. Belkin, R. N. Oddy, and H. M. Brooks. Ask for information retrieval: Part I: background and theory; Part II: results of a design study. Journal of Documentation, 38(2) 61-71 and 38(3) 145-164, 1982.

[Norman 1988] D. A. Norman. The Design of Everyday Things. Doubleday, New York, 1988.

[Azzopardi 2007] L. Azzopardi. Towards evaluating the user experience of interactive information access systems. In Proc. of WISI Workshop at SIGIR 2007, 2007

[Harter 1992] S. P. Harter. Psychological relevance and information science. JASIS, 43(9):602–615, 1992.

[Cooper 1968] W. Cooper. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. American Doc., 19:30–41, 1968.

[Azzopardi 2009] L. Azzopardi. Usage based effectiveness measures: monitoring application performance in information retrieval. In Proceedings of the 18th ACM Conference on information and Knowledge Management, Hong Kong, China, November 2009, 631-640.

[Elliott et al. 2010] D. Elliott, R. Glassey, T. Polajnar, and L. Azzopardi. Finding and filtering information for children. In Proceedings of the 33rd international ACM SIGIR Conference on

Research and Development in information Retrieval, Geneva, Switzerland, July 2010, pages 702-702.

[Glassey et al. 2010] R. Glassey, D. Elliott, T. Polajnar, and L. Azzopardi. Interaction-based Information Filtering for Children. In Proceedings of the 3rd international ACM Conference on Information Interaction in Context, New Brunswick, New Jersey, U.S.A., August 2010.

[Atterer et al. 2006] Richard Atterer, Monika Wnuk, and Albrecht Schmidt: Knowing the User's Every Move - User Activity Tracking for Website Usability Evaluation and Implicit Interaction In Proceedings of The Fifteenth International World Wide Web Conference, Edinburgh, May 2006, pages 203–212.

[Kelly and Teevan 2003] D. Kelly and J. Teevan. Implicit feedback for inferring user preference. SIGIR Forum, 37(2):18–28, 2003.

[Kelly 2004] D. Kelly. Understanding implicit feedback and document preference: a naturalistic user study. PhD thesis, Rutgers University, New Brunswick, NJ, USA, 2004.

[White and Kelly 2006] R. W. White and D. Kelly. A study on the effects of personalization and task information on implicit feedback performance. In Proc. of CIKM ’06, pages 297–306, 2006.

[Hamming 1950] R. W. Hamming. Error detecting and error correcting codes. Technical report, Bell System Technical Journal, 1950.