2nd Workshop on LifeLong User Modelling

Frank Hopfgartner[1], Judy Kay[2], Bob Kummerfeld[2], and Till Plumbaum[1]

[1] Technische Universität Berlin, Germany
[2] University of Sydney, Australia
`http://lifelogging-workshop.org/`

Abstract. Recent technological developments enabled users to create detailed log files containing their everyday activities. These log files, also referred to as lifelogs, open gates for interesting new research challenges, such as the identification of long-term and short-term user interests and behaviour. In the context of this workshop, we focus on an emerging challenge, namely the creation of lifelong user models from a potentially large and diverse data corpus that has been created over a very long period of time. Such lifelong user models pose interesting questions, e.g. regarding the scalability of existing user modeling techniques with respect to the diverse input sources and the pure size of the lifelogs. The importance of this role for the lifelong user model is reflected in the identification of personalised lifelong learning as a Grand Challenge Problem by the Computing Research Association (CRA), United Kingdom Computing Research Committee (UKCRC) and National Academy of Engineering. Following the successful first Workshop on Lifelong User Modelling which was held in conjunction with UMAP 2009 (`http://rp-www.cs.usyd.edu.au/~llum/2009_UMAP09_llum/`), this workshop provided a platform for researchers from both user modelling and lifelogging communities to discuss emerging research trends in this field.

1 Introduction

Nowadays, we are surrounded by technology that assists us in our everyday life. We use GPS devices to navigate from A to B, we use all kind of sensors to track our sport activities, we query the WWW for information while on the go and we use all kinds of devices and software to communicate with our friends and family and share opinions, pictures, etc. With today's technology, we have the capability to automatically record at large-scale the places that we have been to, things we have seen, people we communicate with and how active we are we're already creating a lifelog. This creation of lifelogs offers new possibilities for personalisation but the resulting data volume raises new challenges. Analyzing this large data corpus will enable us to better understand ourselves: What are my habits and interests? Or, even more specific: Do I live a healthy life? Answering these questions can lead to a more conscious lifestyle. One big challenge is the creation and management of long term, even life long, user models These must capture salient aspects about the user over very long periods of time, possibly spanning periods from early childhood into old age. These models have to handle changing interests over time. Also, such lifelogging models have to be usable by different applications. Other challenges pertain processing big data and identifying user interests, skills etc. and their usage in real world systems like health or recommendation systems.

The workshop improved the exchange of ideas between the different research communities and practitioners involved in the research on user modeling and lifelogging. The workshop will focus on the following key questions:

– What lifelogging techniques exist that can benefit from long-term user modelling?
– How can user interfaces assist to explore lifelogs and/or the underlying user models?
– What personalisation techniques can be used in the context of lifelogging?
– How should privacy issues be addressed when it is possible to create detailed user models covering every aspect of ones life?
– What are the particular representational requirements for life-long user modelling?
– What are the requirements for enabling a life-long user model to be useful for a range of applications?
– Which aspects need to be part of the foundation design of technical solutions that will ensure the user's privacy over their life-long user model?
– How will we ensure users can control and share their life-long user model effectively?
– What are the relevant existing standards that should be part of life-long user modelling and where is there a need for additional standards?

2 Acknowledgements

The LLUM chairs would like to thank all authors for their submissions and the UMAP workshop chairs for the help of the workshop organization. Moreover, we also would like to thank the following members of the Program Committee for their reviews:

– Rami Albatal, Dublin City University, Ireland
– Susan Bull, University of Birmingham, UK
– Benjamin Kille, TU Berlin, Germany
– Federica Cena, Universita degli Studi di Torino, Italy
– Anthony Jameson, DFKI, Germany
– Chad Lane, University of Southern California, USA
– Hyowon Lee, MIT, USA
– Na Li, Dublin City University, Ireland
– Pasquale Lops, University of Bari, Italy
– Gord McCalla, University of Saskatchewan, Canada
– Cecile Paris, CSIRO, Australia
– Alan Said, CIW, Netherlands

The workshop keynote was presented by Dr. Cathal Gurrin from Dublin City University, who talked about "Experience of a Lifelogger: Tasks and Challenges". In the references the papers accepted for presentation at LLUM 2013 are presented [1–3].

References

1. Iyilade, J., Vassileva, J.: A decentralized architecture for sharing and reusing lifelogs. In: Second Workshop on LifeLong User Modelling, in conjunction with UMAP 2013. (2013)
2. Tang, L.M., Kay, J.: Lifelong user modeling and meta-cognitive scaffolding: Support self monitoring of long term goals. In: Second Workshop on LifeLong User Modelling, in conjunction with UMAP 2013. (2013)
3. Šajgalík, M., Barla, M., Bielikova, M.: Efficient representation of the lifelong web browsing user characteristics. In: Second Workshop on LifeLong User Modelling, in conjunction with UMAP 2013. (2013)

A Decentralized Architecture for Sharing and Reusing Life-logs

Johnson Iyilade and Julita Vassileva
Computer Science Department, University of Saskatchewan, 110 Science Place
S7N 5C9 Saskatoon, Canada
{Johnson.Iyilade, Julita.Vassileva}@usask.ca

Abstract. Nowadays, there are growing number of devices and applications that assist user in her day-to-day activities. These devices and applications collect and store enormous amount of information about user's daily lives - clickstreams, events, interests, etc. These life-logs are valuable data source for enriching user modeling when shared and reused across applications and devices. While many frameworks and solutions have been proposed for user model sharing and reuse, most existing solutions follow centralized architectural approach where the data is collected from the various sources and stored on a central server before it is shared with the user-adaptive application. However, centralized server can become a single point of failure and can be a ready target for hackers. In this paper, we describe a decentralized approach for multi-application life-logs sharing and reuse. We outline the essential components of a decentralized user life-logs sharing across applications and services on the web.

Keywords: User modeling, Life-logging, Information Storage, User Data Sharing, Reuse

1 Introduction

The ubiquity of the Internet and advances in mobile and communication technologies have led to a rapid increase in the number of digital devices (e.g. tablets, smartphones, wearable sensors etc) and applications (e.g. blogs, calendars, video, music etc) - designed to assist and automate human tasks and activities, enrich human social interaction and enhance our physical world interaction [1,2]. While interacting with these devices and applications, user leaves enormous amount of digital traces about their daily activities, events, locations, and interests. Although, the idea of keeping a record of our everyday life and activities (so-called *life-logging*) is not new (as it has been explored for the past four decades since the work of Steve Mann in the 1970s [3]), one essential difference today is that while user's life-logs were initially confined to what is gathered by a single application or device in a stand-alone machine or on a local network, user's activities are now being recorded in different

contexts by several independent applications and devices connected to the web. Ability to share, aggregate and reuse information collected by these independent applications are desirable goals for life-logs [4], particularly, to enrich user modeling, since they are valuable sources of information about user's activities, interests and preferences.
Obviously, the idea of sharing, integrating, and reusing user information gathered from many applications or sources is not a new problem in user modeling, many early efforts at addressing this problem, however, follows a centralized approach where the data gathered by the various sources are first integrated and stored in database on a server [5], many servers [6] or on a cloud [7]. Although, it could be argued that a central storage of integrated user model has the advantages of ensuring consistency and availability of data, however, as pointed out in [8], centralized architecture is very restrictive and applications have to adhere to user model representation and language of the central server. In addition, centralized servers often have well-defined points of access which can become the central point of failure. Privacy and security issues are also serious threats to a centralized storage.

In this paper, we describe a decentralized architecture for life-logs sharing, integration and reuse. In decentralized approach, rather than transferring and storing all of user’s life-logs to a central repository for later sharing and reuse, each application maintains autonomy in the storage of data and only shares data that is relevant for a particular purpose of adaptation. Integration also occurs at the point of use. The described architecture support basic middleware services such as source selection, semantic mapping, and data integration.

The remainder of the paper is structured as follows. In Section 2, we provide a brief survey of related work. In Section 3, we describe the essential components of decentralized life-logs sharing architecture, and Section 4, concludes the paper.

2. Related Work

Since the 1990s, there have been various efforts towards providing tools and infrastructure that would facilitate the sharing, reuse and aggregation of user information gathered by different independent applications that interacts with the user. An important issue to be considered, however, is the network architecture that could support flexible and secure exchange of user data – i.e. whether the architecture of the user model should be centralized or decentralized [8]. In a centralized approach, the integrated user model is stored in a central server, many servers or on a server in the cloud [7]. The model is then shared across several user-adaptive applications. The first known centralized server for user modeling is the *generic user modeling server* [5] proposed in the 1990s. Subsequently, a number of frameworks for user modeling that follows the centralized architecture have also been proposed. In [9], a framework called IDIUMS was proposed for sharing data between user models in adaptive applications. IDIUMS provides a centralized storage for user model gathered across applications. User adaptive applications can then reuse the information in the central storage through a RESTful Web Service interface. Bielikova and Kuruc [10] also propose a reusable web service called User Model Web Service (UMoWS) for facilitating user model sharing and reuse. In their approach, adaptive applications communicate via SOAP/HTTP with UMoWS which acts as a store of user characteristics for adaptation. In [11], Personis Server, a centralized server was also proposed for storing and reusing user model. User data is stored on a server locally or in the cloud [7]. The central focus of the Personis Server is to allow for scrutability of adaptive systems. In [12], an architecture was proposed for collecting user information into a central Life-long User Model (LLUM) repository which can then be enhanced with information from external sources for recommendation and modeling. As noted earlier, centralized architecture approach guarantees consistency and availability of data, however, centralized approach often require applications to adhere to user model representation and language of the central server. In addition, centralized servers often have well-defined points of access which can become the central point of failure. Maintaining user data privacy and security is a big challenge, particularly with the growing global rise in data security breaches [13] which makes a centralized storage an easy target for hackers. Any successful hack into the server can grant the hackers access to all of the personal life data of the user, thereby exposing the user to unwanted harms.

An alternative approach to information exchange is a decentralized architecture. Here, fragments of user information/model are kept and maintained by each independent application. Only relevant information is exchanged and integrated as needed to fulfill a particular adaptation purpose. Decentralized approach requires integration [14] and semantic mapping [15] techniques provided by a broker for successful exchange.

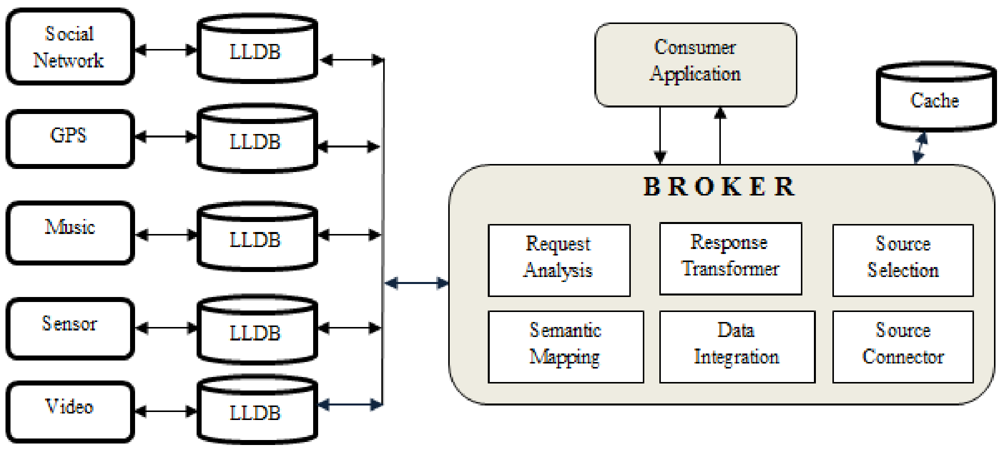

Figure 1: Architecture for Life-logs sharing across applications

3 Life-logs Sharing Across Applications

User Life-logs are currently fragmented in various independent databases, applications and devices. A decentralized architecture that supports the sharing and reuse of the life-logs across applications is presented in this section. The basic components to support decentralized architecture include independent data stores/sources, a broker/aggregator, and the consumer application which needs the data for adaptation purposes. As shown in Figure 1, user interacts with different applications and devices (e.g. social network application, GPS, Music app, Sensor, Video App, etc). These applications and devices collect different kinds of data which are archived in a Life-logs Database (LLDB) maintained by the individual application. Each application might employ various security mechanisms such as encryption to ensure that the data is kept safe. In addition, the application also regulate access to the data using pre-defined privacy policy which determines what part of the data is shared, with whom it is shared and for what purpose.

An important component of this architecture is the data broker. The broker provides middleware services and handles every request for data for personalization by the consumer application. The broker processes the request and communicates with the various data sources to retrieve and aggregate the relevant data. The final aggregated data is not stored by the broker but transmitted to the data consumer for further processing (e.g. reasoning and inference), however, partial results during the aggregation process can be temporarily stored in a cache. The core components of the broker and their functions are briefly described below:

a.) Request Analyzer

The request analyzer takes an incoming request from the consumer application or device in order to determine what the request was and what data is required to fulfill the request. Every request contains the *identity of the consumer application* (for authentication and authorization), the *purpose of adaptation*, the *identity of the user* whose data is being requested. The request can be sent to the broker as HTTP/GET request.

b.) Source Selection

Another important component of the broker is selecting the data sources that have user information relevant to the request based on a specified purpose of use. This is handled by the *source selector*. In an open environment, selection is also important in order to ensure trustworthiness and quality. Hence, trust mechanisms can be employed for determining which provider is to be selected using quality metrics such as *reliability*, *availability*, and *integrity*.

c.) Source connector

After selecting the sources for the data retrieval, different kinds of connection will be required to retrieve the data, particularly when the data is from different sources which requires different application interface for retrieval. For example, for a social network store, a *Social Connector* may be required while for a RESTful resource, an HTTP/GET protocol will be require. The task of the source connector is to determine the appropriate connection mechanism to retrieve data from each source. At this stage also, the source connector may have to perform *user identity mapping* to ensure proper user authentication in order to be able to retrieve the user information from the various sources.

d.) Semantic Mapping

Since the various data sources may use different representation for data it stores, an important task is semantic mapping. The goal of semantic mapping is to resolve the differences in representation from the specific representation of each data sources to a generic representation for the aggregated model and vice versa. In [14], the mapping was done using a hand-crafted mapping rule, while a mediation framework for mapping is presented in [15].

e.) Data Integration

The data integration component is responsible for merging the retrieved data from various sources into user model of proper granularity and resolving all conflicts (e.g differences in data values) associated with the merger.

f.) Response Transformation

The function of the response transformer is to ensure that the data is in the proper format required by the consumer application. For example, this may require the transformation into an RDF format, XML format, FOAF, JSON or any other format required by the consumer application.

4 Conclusion

Our goal in this paper is to describe a generic architecture for sharing and reusing life-logs gathered by many applications and devices. We particularly focus on application-to-application user data sharing environment. A very important issue is whether the architecture will be centralized or not. Although, many of the existing approaches follows the centralized architecture where user data is first integrated and stored on a central repository before sharing. This paper presents an alternative approach to user life-logs sharing and reuse based on decentralized architecture. Each application maintains fragments of user life-logs collected in a particular context while interacting with the user. User modeling, in this case, is then a series of process involving tasks such as request analysis, source selection, semantic mapping, data integration, and response transformation performed by the broker.

References

[1] Poslad, S (2009). "Ubiquitous Computing: Smart Devices, Environments and Interactions". John Wiley & Sons Publishing. ISBN: 978-0-470-03560-3

[2] Seng, L (2007). "Context-aware Pervasive Systems: Architectures for a New Breed of Applications". Auerbach Publications. ISBN: 0-8493-7255-0

[3] Kieron, O., Tuffield, M. M., Shadbolt, N. (2009). "Lifelogging: Privacy and empowerment with memories for life". Identity in the Information Society (Springer) 1 (1): 155. doi:10.1007/s12394-009-0008-4

[4] Kay J. and Kummerfield B. (2009). "Life Long User Modeling: Goals, Issues and Challenges". 1st Workshop on Lifelong User Modelling Workshop held in conjunction with UMAP.June 22-26, 2009, Trento, Italy. Online at: http://sydney.edu.au/engineering/it/~llum/2009_UMAP09_llum/Kay_Kummerfeld.pdf

[5]Kobsa, A.2001. Generic User Modeling Systems. User Modeling and User-Adapted Interaction. Vol. 11, pp. 49-63

[6] Fink, J. and Kobsa, A. 2000. A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web. User Modeling and User-Adapted Interaction. vol 10. pp 209-249.

[7] Dolog, P., Kay, J. Kummerfeld B. (2009). Personal Lifelong User Model Clouds. Proceeding of the Lifelong User Modeling Workshop at UMAP09, June, 2009, Trento, Italy, Pp: 1-8

[8] Cena F. and Gena C. (2012). Designing and evaluating new generation user models (NewGUMs). Tutorial at UMAP 2012. Slides Accessed Online at: http://umap2012.polymtl.ca/en/workshops-and-tutorials/84-workshops-and-tutorials/104

[9] Prince R. and Davis H. (2011). IDIUMS: sharing user models through application attributes. Poster presentation, Proc. User Modeling, Adaptation and Personalization, UMAP 2011, Girona, Spain, July 2011. Springer Verlag. Berlin-Heidelberg, ISBN 978-3-642-22362-4. pp 40-42

[10] Bielikova M. and Kuruc J. 2005. Sharing User Models for Adaptive Hypermedia Applications. Proc. 5th Int. Conf. Intelligent Systems Design and Applications. Washington DC, USA. pp. 506 - 513.

[11] Kay J., Kummerfeld B., Lauder P. 2002. Personis: A server for user models. In P. de Bra, P. Brusilovsky, and R. Canejo, editors. Proc. Adaptive Hypermedia and Adaptive Web-based Systems, Springer Verlag. ISBN:3-540-43737-1. pp. 203-212.

[12] Elliott D., Hopfgartner, F., Leelanupab, T., Moshfeghi, Y. and Jose J.M. (2009). An Architecture for Life-long User Modelling. Proceedings of the Lifelong User Modelling Workshop, at UMAP'09 User Modeling Adaptation, and Personalization. pp 9-16

[13] Hartwig, R.P. and Wilkinson C. (2013). "Cyber risks: The Growing Threats". Accessed Online at: http://www.iii.org/assets/docs/pdf/paper_CyberRisk_2013.pdf. Last accessed date: 08-April-2013

[14] Fabian Abel, Nicola Henze, Eelco Herder, Daniel Krause. Linkage, Aggregation, Alignment and Enrichment of Public User Profiles with Mypes. Proc. 6th Int. Conf. Semantic Systems (I-SEMANTICS), Graz, Austria, September 2010, Article No. 11. ISBN: 978-1-4503-0014-8.

[15] Berkovsky S., Kuflik T., Ricci F. 2007. Mediation of User Models for Enhanced Personalization in Recommender Systems". User Modeling and User-Adapted Interaction. Vol. 18, Issue 3., pp 245-286

Lifelong User Modeling and Meta-cognitive Scaffolding: Support Self Monitoring of Long Term Goals

Lie Ming Tang and Judy Kay

University Of Sydney
http://chai.it.usyd.edu.au
ltan8012@uni.sydney.edu.au

Abstract. It is becoming increasingly easy to capture lifelong data about a person. This creates the potential for valuable new ways for people to self-monitor diverse aspects their lives. If people are to do this effectively, we need to ensure that relevant aspects of lifelong user models are available to the user, in a form that is meaningful. Beyond this, current research indicates that most people need a meta-cognitive scaffolding to make more effective use of such information. This paper reviews work about such meta-cognitive scaffolding. Then it presents use cases illustrating how this scaffolding can help people achieve long term goals. We use these to identify requirements for the architecture and associated user interfaces for a lifelong user model with scaffolded self monitoring. Our key contribution is the requirements for a system that scaffolds self monitoring of long term goals, based on data stored in a lifelong user model.

Keywords: meta-cognition, scaffolding, self monitoring, lifelong user modeling, goal setting

1 Introduction

Computing devices have rapidly advanced in processing power, mobility and communication. We can now readily collect very detailed personal data, for aspects such as physical activity over long periods of time (e.g., steps per day, over weeks, months and years). It is also easy to store vast and growing amounts of such information. This constitutes a form of lifelong user model. When this is suitably interpreted and relevant sets of information are combined, lifelong user modeling has the potential to help people achieve their long term goals such as healthy weight, increasing physical activity [13] or lifelong learning [18] such as learning a foreign language. Such user models can play a key role in behavioural change [20] and self regulated learning [4].

It is important for people to monitor and reflect on their progress, adjust their actions and behaviour based on real evidence over the long term. Granular and long term data about individuals can be captured in lifelong user models and help accurately measure performance, progress and relate to their long term goals.

The interfaces through which people see their lifelong user models are critical as they encourage trust and credibility [10] in behaviour change applications.

This paper is concerned with helping people make effective use of displays of their lifelong user model and for goal setting. In particular, we take account of research which indicates the need to *scaffold* users in the *meta-cognitive processes* of *self monitoring* and *reflection*. Such scaffolding has been effective in computer based learning [2] and in supporting physical activity [17, 14] They are likely to be key in enabling users to take on the role of end-user programmers, controlling what data are collated in lifelong user models and then interpreting it in terms of 'means' goal intended to enable long term 'end' goal as termed in [5].

Consider a scenario where a hypothetical user, Alice, has an *end goal* to maintain the recommended level of moderate physical activity, at least 30 mins / 5 days a week [13] . She sets *means goals* of: walking twice a week and jogging once a week. If suitable data collection about these activities is available, she can see whether she is achieving each means and end goal. This paper explores the requirements for an interface that can facilitate self monitoring and reflection. We draw on lifelong user modeling [16, 15], meta-cognitive principles of self monitoring [23], reflection [11] and computer based meta-cognitive scaffolding support [4].

The next section reviews key related work. Then we present a set of use cases as a foundation for our identification of requirements for system design and interfaces that scaffold self monitoring for reflection to support long term goals.

2 Related work

This section introduces the three key foundations for our work. The first of these is lifelong user modeling. The second is work on meta-cognition and the need for scaffolding. We then introduce open learner modelling since it draws the earlier two parts together.

Long term user modelling refers to the capture of long term information about the users [16, 15]. This emerging view of a user model diverges from the classical works where a user model was typically tightly integrated into a single application. There are many challenges in lifelong user modelling, (see [15] for a review of these). For this work, the key concern is that the model can hold a large amount of information that has the potential to be valuable for people to monitor their progress towards their long term goals.

Meta-cognition refers to knowledge about one's own knowledge [22]. It includes diverse skills, such as the dozens identified in [4]. For our work, the key skills relate to reflection that is based on long term user models. To achieve long term goals, a person draws on meta-cognitive skills in setting goals, thinking about how realistic they are, monitoring progress [11] and using this to revise goals, recognising that some were achieved, some were not and that some were unrealistic [23]. Effective interpretation of the information, in order to monitor progress is critical. Yet many people do not automatically make good use of information about themselves to self-monitor [3, 21]. However, there is evidence that systems which do scaffold metacognitive skills, such as self monitoring and reflection, can improve these critical skills [2]. Open learner models (OLMs) provide interfaces that enable people to view, and sometimes, to manipulate their user model [6]. For this paper, OLMs provide a body of work on interfaces that externalise a learner model. Some OLMs have tackled the challenge of showing large user models in a cognitive domain, as in VLUM [25] and this enabled learners to assess their progress [15] Another presented a visualisation to support reflection [12] through visualisation of student activity relative to other students.

Ambient displays can provide a form of OLM that is designed to blend into the environment until the user wants to consult it. These may be important for people monitoring long term goals. For example, *Breakaway* [14] was designed to help people remember to achieve their goals to take breaks from long periods of sitting. A small sculpture based on user's chair appears upright if the user had taken breaks and it slouched if the user had been sitting for a while. The CareNet glanceable display on a picture frame [9] showed key information about an elderly person's daily activity. This was a form of OLM for their carer family living elsewhere.

While results are not conclusive due to sample size and technology, using visual abstraction techniques to present information is promising. A study in the open learner model for children [6] represented a child's knowledge state as pictorial abstraction of misconceptions where a tree is healthy or dying the more misconception a child has on a topic. An example of glanceable display is the UbiFit system [8] showing a garden where number of flowers represent different types of activities. There are also studies where social interaction is tested to encourage physical activity [19].

While there has been some work that has tracked use of OLMs over months (e.g., [6]), the challenge of supporting self monitoring and reflection over years and decades is new ground for lifelong user modelling.

3 Use Cases of scaffolded self monitoring based on a lifelong user model

In this section, we describe three use cases related to a hypothetical person 'Alice'. It will present examples of visualisations and discuss potential for customisation and personalisation. It will also identify how we can scaffold self monitoring and reflection to achieve long term goals.

Use Case 1 : To run City2Surf (14 km) in 1 hour and to maintain at least 100 mins of exercise per week
It is recommended for adults to have a minimum of 30 min of moderate-intensity aerobic physical activity 5 times per week [13]. Alice would like to maintain this and also to train for City2Surf which is an annual 14 Km fun run in Sydney. She decides to track running and walking exercises where she uses a pedometer with time recording capability [1] and stopwatch to time her runs.

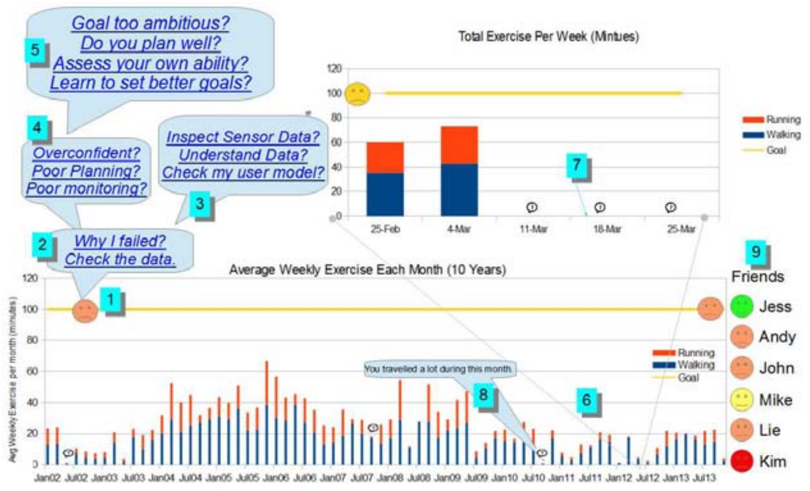

Fig. 1. Graph showing average weekly exercise 2002-2013 Walking & Running and includes some example interface elements.

Figure 1 shows elements that monitor Alice physical activity over time. The Goal line (yellow) represents her end goal of 100 minutes exercise per week and allow her to reflect on her achievement and drill down into a particular month (label 6). The *"smilie"* face is an example of physical abstractions of goal attainment label 1 inspired by [8, 17]. Elements of meta-cognitive scaffolding can be incorporated in labels 2, 3, 4, and 5 where users are asked to reflect on their progress, understanding and meta-cognitive skills. Label 8 illustrates how system represent knowledge about a user e.g., Alice travelled in Jul-10, got injured during Aug-10 which can be incorporated into the user experience to foster reflection in her ability, accuracy of data and goal setting.

Figure 2 shows another view focusing on the goal of running City2Surf 14 km in 1 hour. Alice can deduce that her speed is reducing over last 10 years. This can help her reflect on whether she has set her goals too high or if other factors such as gaining weight and being less active could have contributed. Meta-cognitive scaffolding can help her through encouraging her to reflect on her ability to seek help, plan and set better goals. By showing her data over a long period of 10 years, she is able to see the truth about her declining ability over time and reflect accurately about her abilities.

Use Case 2 : Avoid prolonged periods of sitting to less than 8 hours per day. Prolonged sedentary behaviour is detrimental to long term health [26]. To monitor this, a combination of devices are employed such as fitbit, desktop PC application, lifelog camera or a seat sensor. It is not always straight forward to determine inactivity due to sensing limitation or user preference so data may not

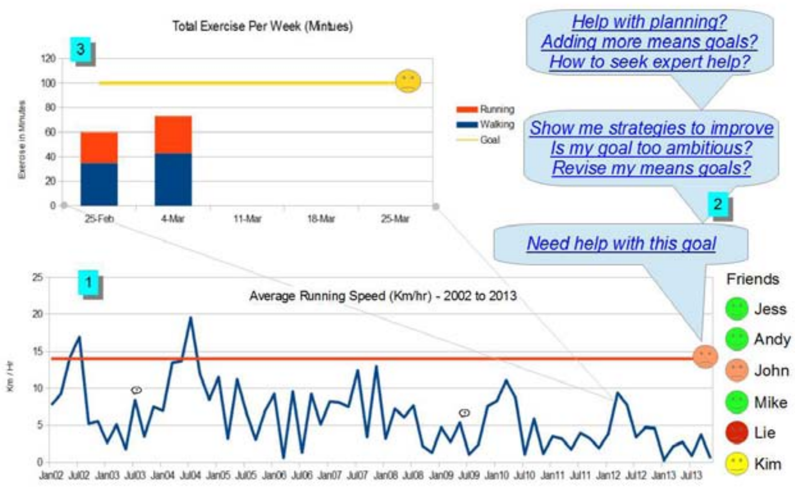

Fig. 2. Graph showing average running speed over 10 years period from 2002 to 2013.

be complete e.g., not all devices are waterproof so person needs to take them off while swimming. A benefit of combining data is that multiple sensors data may be complementary. For example, a desktop PC application detects computer use and seat sensor records sitting on a chair and both detects user may be sitting. However, a person may be sitting and not use computer or use computer and not sitting on a smart chair. When combined, these sensors can provide a better longitudinal view of user's sitting behaviour. Lifelong user modeling can facilitate consolidation of data from different sensors and provides foundation to improve accuracy of inferences. In addition, users may want to analyse the data over the long term to see patterns of behaviour. Figure 3 shows how inactivity data can be presented to show breakdown such as driving, computer use and watching TV. Combined with interface elements introduced in Use Case 1 Figure 1 and figure 2, there is potential to further personalise interfaces to make sense of data.

Use Case 3 : Learn at least 2000 common usage words and passing language proficiency test.
In this scenario Alice is not a native English speaker and sets the long term goal of learning at least 2000 words in common usage. Lifelong learner model [15] can allow applications to capture data and scaffold for Alice to reflect on her progress and learning strategies. Figure 4 plots the number of words she learnt over time versus the mean lower and upper bound of other learner's progress. It shows her progress is only slightly better than lower bound indicating she should review her learning strategy for improvement. Combined with potential scaffolding elements illustrated in use case 1 figure 1 and figure 2, we can scaffold reflection and improve long term learning.

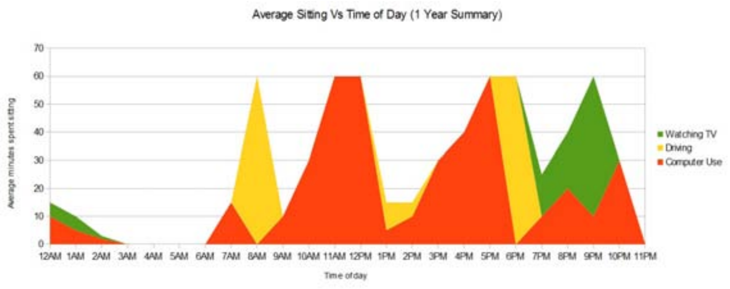

Fig. 3. Chart showing average sitting time over time of day (1 Year).

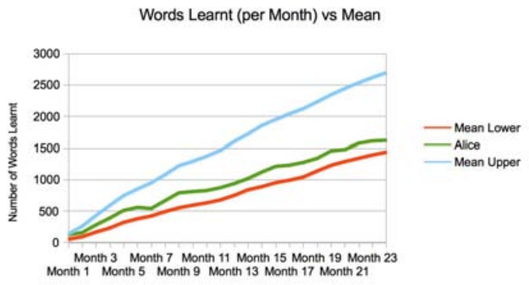

Fig. 4. chart showing average sitting time over time of day in minutes.

4 Requirements for a lifelong user model with scaffolding for self monitoring and reflection

Lifelong user model provides us the foundation to support meta-cognitive scaffolding for self monitoring, reflection and goal setting. We have identified a number of use cases and illustrated how some of these tools can be utilised to achieve this. In this section we describe requirements for future systems to effectively utilise lifelong data and to support long term goals. We present these requirements under three perspectives.

4.1 Capture and present data in a way users can use effectively.

From the user's perspective, to utilise vast amounts of personalised data over long term and how they relate and contribute towards one's long term goals is a daunting proposition.

☐ **Capture data over the long term.**
It is important that data is captured and kept over long term as it would allow us to monitor progress, reflect on user behaviour and provide better personalisation.

☐ **Capture data from different devices.**
Data can come from an ecosystem of devices (e.g., fitbit, desktop, mobile phone, fixed cameras) and a system should support capture and combining of this data.

☐ **Address key personalisation problems.**
Future systems needs to address problems of *privacy, invisibility of personalisation, errors in user model, wasted user model* and *control* as described in [16]. Key concepts being users need to be aware and take control of what system knows about them and how it uses this information (*invisibility, control*), avoid silos of data (*wasted use model*) and control privacy.

☐ **Help people interpret data.**
It is not sufficient to simply capture data but also turn it into useful information. Visualisations and interfaces should allow users to answers questions such as have they met their goals, reasons for failure and how to improve. Interfaces should be easy to interpret yet provide enough capability to perform analysis. An approach might be to provide meta-cognitive scaffolding based on lifelong user model data (e.g., level of computer literacy, academic ability etc.) to improve meta-cognition. An example of a framework that can support this is the Personis platform described in [15]. The concept of separating the lifelong user model with applications and views allow us to better personalise interface.

4.2 Personalisation and control of data.

It is important for system to capture user preferences and how they want to interact with their own data and user models. A system should also follow certain guidelines to support user interaction and visualisations.

☐ **Personalisation of data and user interfaces**
Different people have preferences for how they want to interact with their data [5] so it is important to allow for a certain level of control.

☐ **Aggregated and abstracted views.**
When dealing with lifelong user data that represents a large dataset it becomes necessary to provide an aggregated views [7] or abstracted views [17]. They can also be unobtrusive and potentially motivational [14]. A system can also provide interactive capabilities to allow users to drill down for more detailed or localised view of the data

☐ **Controlling how you view your data.**
A system should provide capabilities to control the level of abstraction and aggregation, where and how to display data, how information can be shared. 'Alice' may choose to show daily walking and jogging data with her personal trainer but only an abstracted *"smilie"* face of goal achievement with friends. System should enable users to scrutinise [15] their user model. E.g., system

decide not to show weight data over time as she never looked at this information in the past. However, upon reflection of her jogging performance, Alice decides that she does want to know about her weight.

☐ **Trust, confidence and coherence of data.**
It is important to foster trust and confidence in a system [10] to support and motivate users. Interface should not mislead or be misinterpreted otherwise it may result in loss of confidence. For example, Alice only wears her fitbit sensor during exercise and also sometimes forgets to wear it. System should indicate not all data is available and prompt for input or correction.

4.3 Support long term goals.

Our key objective is to support users in meeting their long term goals. The list of questions below identifies the key challenges a system should aim to support.

☐ **Set new and better long term goals.**
Elements of the system can provide meta-cognitive scaffolding to support goal setting. For example, system can tell Alice that she is usually too overconfident and she may be too ambitious with 14km in 1 hour and teach her to improve her ability in setting goals for the future.

☐ **How to achieve long term goals.**
System can provide scaffolding to help set means goals to achieve their long term end goals. For example, Alice sets her long term goal of running City2Surf 14 Km in 1 hour within 1 year. System can draw on her current routines to suggest a program of means goals to incrementally increase the speed and distance of her regular jogging over the period of 1 year.

☐ **Monitoring of long term goals.**
Scaffolding should support monitoring of progress and reflect on whether users have achieved their long term goals. Users should be allowed to scrutinise goals, inferences and seek help if required.

☐ **Maintaining motivation over long term.**
Systems that targets to help users achieve long term goals should aim to provide some kind of motivation and support mechanism over long term. For example, this may be achieved through providing accurate information, timely reminders [8] and social encouragement [24] and can be based on the user lifelong user model.

5 Discussion and conclusions

Growing and widely available pervasive devices, storage and communication allow us to collect and store very granular and personal data over the long term. In this paper, we discussed this through 'Alice' a hypothetical person who has different types of long term goals and how she can combine different data and example interface elements to achieve this. We discussed lifelong user modeling [15], meta-cognitive scaffolding techniques [4] and interface designs that can be applied to this domain of long term goals. Finally, it presented the requirements for systems as guidelines for how future systems and studies can utilise these techniques. Suggested future work include performing evaluation on these techniques and their strength and weaknesses. We can also assess suitability for different use cases, scenarios and types of long term goals.

References

1. fitbit. http://www.fitbit.com/. Accessed: 2013-04-01.
2. Roger Azevedo, Jennifer G Cromley, Daniel C Moos, Jeffrey A Greene, and Fielding I Winters. Adaptive content and process scaffolding: A key to facilitating students self-regulated learning with hypermedia. *Psychological Testing and Assessment Modeling*, 53:106–140, 2011.
3. Roger Azevedo, Jennifer G Cromley, and Diane Seibert. Does adaptive scaffolding facilitate students ability to regulate their learning with hypermedia? *Contemporary Educational Psychology*, 29(3):344–370, 2004.
4. Roger Azevedo, Amy Witherspoon, Amber Chauncey, Candice Burkett, and Ashley Fike. MetaTutor: A MetaCognitive tool for enhancing self-regulated learning. In *Proceedings of the AAAI Fall Symposium on Cognitive and Metacognitive Educational Systems*, pages 14–19, 2009.
5. Debjanee Barua, Judy Kay, and Cécile Paris. Viewing and Controlling Personal Sensor Data: What Do Users Want? In Shlomo Berkovsky and Jill Freyne, editors, *Persuasive Technology*, volume 7822 of *Lecture Notes in Computer Science*, pages 15–26. Springer Berlin Heidelberg, 2013.
6. Susan Bull and Judy Kay. Open Learner Models. In Roger Nkambou, Jacqueline Bourdeau, and Riichiro Mizoguchi, editors, *Advances in Intelligent Tutoring Systems*, volume 308 of *Studies in Computational Intelligence*, pages 301–322. Springer Berlin Heidelberg, 2010.
7. Stuart K Card, Jock D Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.
8. Sunny Consolvo, Predrag Klasnja, David W. McDonald, and James A. Landay. Goal-setting considerations for persuasive technologies that encourage physical activity. In *Proceedings of the 4th International Conference on Persuasive Technology - Persuasive '09*, page 1, New York, New York, USA, April 2009. ACM Press.
9. Sunny Consolvo, Peter Roessler, and BrettE. Shelton. The CareNet Display: Lessons Learned from an In Home Evaluation of an Ambient Display. In Nigel Davies, ElizabethD. Mynatt, and Itiro Siio, editors, *UbiComp 2004: Ubiquitous Computing*, volume 3205 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2004.
10. B J Fogg. *Persuasive Technology: Using Computers to Change What We Think and Do*, volume 5 of *The Morgan Kaufmann series in interactive technologies.* Morgan Kaufmann, 2003.
11. Claudia Gama. Metacognition in interactive learning environments: the reflection assistant model. In *Intelligent Tutoring Systems*, pages 668–677. Springer, 2004.
12. Sten Govaerts, Katrien Verbert, Erik Duval, and Abelardo Pardo. The student activity meter for awareness and self-reflection. In *Proceedings of the 2012 ACM*

annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts - CHI EA '12, page 869, New York, New York, USA, May 2012. ACM Press.

13. William L Haskell, I Lee, Russell R Pate, Kenneth E Powell, Steven N Blair, Barry A Franklin, Caroline A Macera, Gregory W Heath, Paul D Thompson, Adrian Bauman, and Others. Physical activity and public health: updated recommendation for adults from the American College of Sports Medicine and the American Heart Association. *Medicine and science in sports and exercise*, 39(8):1423, 2007.
14. Nassim Jafarinaimi, Jodi Forlizzi, Amy Hurst, and John Zimmerman. Breakaway: an ambient display designed to change human behavior. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1945–1948, New York, NY, USA, 2005. ACM.
15. J Kay and B Kummerfeld. Lifelong learner modeling. *Adaptive Technologies for Training and Education. Cambridge University Press, New York*, pages 140–163, 2012.
16. Judy Kay and Bob Kummerfeld. Creating personalized systems that people can scrutinize and control: Drivers, principles and experience. *TiiS*, 2(4):24, 2012.
17. Tanyoung Kim, Hwajung Hong, and Brian Magerko. Design requirements for ambient display that supports sustainable lifestyle. In *Proceedings of the 8th ACM Conference on Designing Interactive Systems*, DIS '10, pages 103–112, New York, NY, USA, 2010. ACM.
18. Christopher Kay Knapper. *Lifelong learning in higher education*. Routledge, 2000.
19. J Maitland, S Sherwood, L Barkhuus, I Anderson, M Hall, B Brown, M Chalmers, and H Muller. Increasing the Awareness of Daily Activity Levels with Pervasive Computing. In *Pervasive Health Conference and Workshops, 2006*, pages 1–9, 2006.
20. D.B.a Portnoy, L.A.J.b Scott-Sheldon, B.T.a Johnson, and M.P.b Carey. Computer-delivered interventions for health promotion and behavioral risk reduction: A meta-analysis of 75 randomized controlled trials, 1988-2007. *Preventive Medicine*, 47(1):3–16, 2008.
21. Ido Roll, Vincent Aleven, Bruce M McLaren, and Kenneth R Koedinger. Improving students help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*, 21(2):267–280, 2011.
22. H Lee Swanson. Influence of metacognitive knowledge and aptitude on problem solving. *Journal of educational psychology*, 82(2):306, 1990.
23. Sigmund. Tobias, Howard T Everson, and College Entrance Examination Board. *Knowing what you know and what you don't : further research on metacognitive knowledge monitoring*. College Entrance Examination Board, New York, 2002.
24. T Toscos, A Faber, K Connelly, and A M Upoma. Encouraging physical activity in teens Can technology help reduce barriers to physical activity in adolescent girls? In *Pervasive Computing Technologies for Healthcare, 2008. PervasiveHealth 2008. Second International Conference on*, pages 218–221, 2008.
25. J Uther and J Kay. VlUM, a web-based visualisation of large user models. *User Modeling 2003*, pages 145–146, 2003.
26. Hidde P van der Ploeg, Tien Chey, Rosemary J Korda, Emily Banks, and Adrian Bauman. Sitting time and all-cause mortality risk in 222 497 Australian adults. *Archives of internal medicine*, 172(6):494–500, March 2012.

Efficient Representation of the Lifelong Web Browsing User Characteristics

Márius Šajgalík, Michal Barla, Mária Bieliková

Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technology
Slovak University of Technology, Bratislava, Slovakia
{sajgalik,barla,bielik}@fiit.stuba.sk

Abstract. Client-based user modelling has already been studied and clearly has its place among generic approaches to the user modelling. It is especially advantageous for lifelong user modelling as it can support the modelling in any time and any place including consideration of user privacy. Emergence of web browser extensions opens up possibilities of pure browser-based realisation of client-based user modelling. In this paper, we focus on the efficient representation of a generic user model inside a web browser, which forms the core part of browser-based user modelling framework in form of a browser extension. Efficiency is crucial also from the lifelong perspective. We propose an efficient method of lifelog indexing and modelling various user characteristics inside the web browser. We evaluated properties of proposed representation and describe its applicability in some common use cases.

1 Introduction

In this modern era of ubiquitous computers in our everyday life, the need of ubiquitous lifelong user modelling approaches has emerged. With regard to the shift in human-computer interaction from desktop computing to mobile, the decentralised client-based approaches have been encouraged to support variety of adaptation goals. From this ubiquitous perspective, the web browser seems to be an ideal choice for lifelong user modelling, since we use it everywhere, every day, on each device – be it a desktop computer, laptop, tablet or mobile phone. The emergence of new web technologies like HTML5 and support for powerful extensions makes the web browser a capable platform suitable to perform user modelling and personalisation across the Web, while still keeping our user profile under our complete control, without the need to disclose our identity, browsing history or our user model to any third-party service.

From the user modelling perspective, we have a tremendous choice in regards of user actions within a web browser. Not only we get a complete surfing history, but we get also the context of all actions such as mouse movements or other tabs opened at the same time. On the other hand, being on a client side, we need to focus much more on the efficiency of all user modelling processes and of all data structures used to capture our user model, since we are rather limited in resources compared to server.

We developed BrUMo[1] – a specialised web browser extension, which represents a browser-based instance of our user modelling and personalisation framework. According to statistics recorded within BrUMo, the average number of visited webpages can reach over 200 per day for some users. It is important to note that nowadays web pages often do not reload the entire content, but only its part is updated via AJAX. If we count all requests made by a single user, the average number of AJAX requests can grow up to over 600 per day. If we are to extract and store keywords or other metadata of each visited webpage and update it according to all changes made within a single visit to index user knowledge or interests, the efficiency is really crucial. We need not only an efficient storage of the model in terms of required allocated space and memory, but we also need to consider efficiency of perpetual retrieval of information from the model.

In this paper, we propose mechanisms to index various user characteristics captured in the user model in an efficient manner. We have extended the concepts of some fundamental data structures like Patricia trie to design a user and a domain interest tree, which provide two perspectives of the user model. Although the usage of such basic data structures as well as their extensions is much broader than its application in user modelling, we consider it very important to deal with the efficient representation of user characteristics in the web browser as its resources and possibilities are still limited there in comparison with server-based or cloud-based solutions.

2 Related Work

In [4] an analysis of multiple challenges of cloud-based user modelling in favour of clients is presented. The authors propose to decentralise and push user models down to the client side and describe a sample PaaS (Personalisation as a Service) architecture. In [1] authors discuss the possibility of scenario where server is used only to update the profile and to perform personalised response computation, while all information is stored locally. The authors of [7] present an architecture of a client-side framework to provide adapted content. However, they use additional storage and management servers. There is also other user modelling framework called PersonisJ [3], which is specially aimed at Android phone platform. There is also an evaluation of its efficiency, which shows how it is important and thus also supports the importance of our work. An approach to web search personalisation described in [11] is based on a model of user interests coming from the user's web browsing history. It is realized as Mozilla Firefox extension, but only to collect the information about user. All computational work is done on the server. However, the author in [11] states the possibility of implementing it solely client-side, which would avoid the privacy concerns arising with server-based approaches.

An exhaustive overview of what data can be captured directly in the browser is analysed in [14]. Another study in [5] shows what additional information about the user can be captured on client side by means of emerging Web 2.0 technologies. GINIS

[1] Browser-Based User Modelling and Personalisation Framework – http://brumo.fiit.stuba.sk/

framework [16] is one representative of browser-based systems. It is a customised tabbed Internet Explorer browser using .NET framework. Although not explicitly stated, we assume they used MS SQL database to log user actions, which is supported by their claim that "it is easy to log high-granularity data using the provided .NET framework". From our perspective, GINIS framework has several disadvantages. It requires user to install and use some non-standard specialised browser, which for example does not guarantee to receive important updates unlike the standard official releases. Moreover, they focus just on their single goal (to classify content as interesting or not) and build a decision tree based on user behaviour data. They do not present any general user model or other mechanisms to index their raw logs. Somewhat more relevant to our work is an example of browser-based user modelling system in form of an extension presented in [8]. There, authors mention that they use HTML5-based SQL database to store two tables – one for keywords with their corresponding frequencies and second for the visited webpages with their visit frequencies. Although this approach has more general user model and all the advantages of browser extension, it fails to provide some additional information about user, which can be required in context of lifelog creation as well as in some common personalisation scenarios.

Yet another approach is to place the user modelling platform to the middle between the Web and a client in a form of specialised proxy. For instance, PeWeProxy is a proxy server, which builds a term-based user model representing the user's interests from keywords and terms automatically extracted from web pages passing through the proxy. It shifts the personalisation part to the client using personalisation scripts embedded into the browsed web pages [10]. Various research extensions like estimation of user interests [6] and web search disambiguation [9] have already been developed for it, though all of them are also run within the proxy server.

To summarise, it seems that there is a demand for pure client-side solutions to user modelling and personalisation, which address the problem of efficient representation of generic lifelong user model. Servers satisfy most of computational power and storage force requirements. Shifting to client however, requires more focus on this topic, since the possibilities and resources are generally much more limited.

3 Representing the User

Web (search) history analysis and keyword-based user models are becoming more and more popular solutions for user modelling [2]. Keywords representing users' interests are relatively easy to acquire and they could be easily presented to a user (to justify or explain personalisation, to enable a user to scrutinise her model and provide an explicit feedback upon it). At the same time, their lightweight semantics provides a solid basis for personalisation. A nice example can be found in [13], where author uses a lightweight folksonomy, which can be considered as a form of collaborative web surfing history, to infer similarity among users or visited documents.

We use keyword-based representation to express various user characteristics such as interests, knowledge, goals, context of work, etc. For example, user interests are represented as weighted vector of terms, where each term is linked to some URL address recorded in the user browsing history. There are multiple terms for each URL and similarly each term can be linked to multiple URLs. These links connecting terms with URLs are also weighted according to their mutual relevance. They denote the relevance of a web page at some URL to given term. Since the term represents a user interest, we can find out how interesting particular web page is by following the corresponding link. It is important to note the variability of terms that can stand for not just words extracted from read articles, but possibly other units like stems, lemmas or concepts. Similar logical representation can be applied to other user characteristics, e.g. terms represent knowledge concepts or particular goals in educational domain.

Currently, in our BrUMo platform, we use keyword extraction to infer user interests. To extract keywords from webpage, we combine multiple methods. First, we consider the content of webpage. We extract the article using Readability[2] and utilise browser's built-in functionality[3] to obtain raw text. In further pre-processing we tokenise the text into words using jspos[4] lexer, filter out stop-words and all words shorter than 3 characters and consider further only nouns as tagged by jspos POS tagger. With these feasible words extracted, we compute relevance of each word as an average of normalised TF-IDF [15] and TextRank [12]. We normalise the TF-IDF value by text length. Afterwards, we look at keywords meta-tag in HTML structure and propagate these keywords by doubling their relevance value. The IDF values were obtained from Google N-gram corpus[5].

In this paper, we present two basic data structures to index user characteristics – user interest tree and domain interest tree. The user interest tree serves perfectly for indexing global user interest. However, a user can have different preferences in different domains. These are represented by so called local interests, since they are significant only within a particular domain. Domain interest tree is designed to index these local interests.

3.1 User Interest Tree

User interest tree captures global user interests. By storing the terms representing those interests using a Patricia trie, we can easily execute all queries based on the term index, e.g., fast insertion/deletion or to iterate over all the terms stored in the tree in alphabetical order. To enable retrieval of the topmost relevant terms to get the most relevant global user interests we extended the basic structure using a labelling technique which enables us to speed up the tree node traversal in an order of term relevance.

Figure 1 depicts an example of such a labelled tree. It stores words and their relevance (in brackets) PEACE (2), PENCIL (3), PEWE (5), SEBE (4) and SET (0). We label each sub-tree by the relevance of the most relevant term in it. Thus, we can easily

[2] Readability – https://code.google.com/p/arc90labs-readability/

[3] textContent property – http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/core.html#Node3-textContent

[4] JavaScript part-of-speech tagger - http://code.google.com/p/jspos/

[5] Google N-gram corpus – http://storage.googleapis.com/books/ngrams/books/datasetsv2.html

retrieve the most relevant term in each sub-tree by following the path labelled by maximal value. The creation of such a tree is simple. The words are inserted in a common manner like into an ordinary Patricia trie. In addition to that, we update all vertices on the path from root to the inserted leaf node so that the above stated labelling rule holds true.

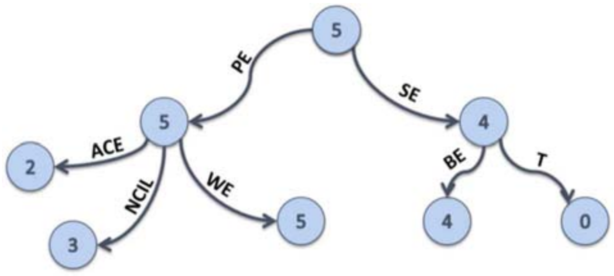

Fig. 1. Labelled Patricia trie

With such a labelled tree we propose following steps to iterate over all of the terms of a valid user interest tree in order of their relevance:

1. Initialise empty array of results and two heaps.
2. Initialise first heap by inserting the user interest tree root node in it. This heap always pops out the node labelled by maximal value.
3. Initialise empty second heap. This one always pops out the most relevant node.
4. Pop out node `v` from first heap. If it is empty, we have already traversed all nodes.
5. If `v` represents a term (not just prefix), push it into the second heap.
6. While there are nodes in the second heap with relevance not lower than label of `v`, pop them into the results array. If the desired count of results is reached, stop here.
7. Push all children of node `v` into the first heap and continue with step 4.

Every node is at worst once inserted and removed from each heap. Time complexity is therefore $O(n \times m \times \log_2(n \times m))$ where n is the number of terms in the tree and m is the length of the longest term, which is asymptotically optimal. However, the advantage of this algorithm over a simple sorting of all terms is the ability to terminate it prematurely once we got the required number of terms. This can greatly reduce the running time of retrieval of just first k most relevant terms to $O(k \times m \times \log_2(k \times m))$. Note that node relabeling is done with insertion/deletion in $O(m)$. Another advantage is that we can use multiple different labels, so that we can retrieve also the most recently added term to get the context of user's work. Additional labels are also important for managing the tree over longer time period and limit the overall tree size. Labels for the least relevant or the oldest terms can be used to remove such surplus terms from the tree since that could mean that user is interested in them no more.

3.2 Domain Interest Tree

Domain interest tree is similar to a user interest tree in its structure, but it is extended with some concepts of the generalised suffix tree. Unlike the generalised suffix tree, the domain interest tree does not need to store all the suffixes of strings, but only those suffixes that represent some subdomain. Thus, primary key in this tree is a URL address and its suffixes corresponding to different subdomains. Domain interest tree can also be labelled like in the case of the user interest tree, e.g., for each sub-tree, its root is labelled with a frequency value of the most frequent URL address within this sub-tree. Thus, we can easily determine the most frequent URLs.

In addition, we store first *k* most relevant interests (terms) in each node within the sub-tree rooted at this node. This allows us to retrieve efficiently local interests within various subdomains. Constant *k* represents a trade-off between performance and precision. Higher *k* means that more terms are propagated to the parent node for the price of lower performance. If we need more than first *k* most relevant terms in a particular subdomain, we can recursively nest further to search the child nodes' *k* most relevant terms until we reach the desired number of a user's local interests.

3.3 Lifelong perspective

Despite the powerful built-in labelling technique, which is used to manage the tree and limit its size over longer time period, it is just not enough from the lifelong point of view. A single tree as it is, can be managed only within given time sliding window. The need of limiting the tree size arises from the fact that we are limited in memory and the naïve idea of simply building one huge tree out of the whole browsing history is indispensable.

Therefore, we propose to factorise the whole browsing history into multiple trees. In other words, we maintain one tree sliding over time and in regular time intervals, take a snapshot of it and store it into database. Moreover, we build a tree hierarchy out of these trees (user/domain interest trees) in order to get the most relevant terms for different time intervals at different abstraction level. We show a sample of such hierarchy in Figure 2. To create a tree on higher level of abstraction, we simply combine content of all underlying trees, but limit the size of the resulting tree to contain just the most relevant features. Using this tree hierarchy, we can easily get the overall lifelong-spanning characteristics, which are stored in the root.

4 Application and Evaluation of User Model

We designed our indexer with its underlying indexing data structures to achieve the best possible time and memory complexity for all needed operations to be ready for real-world lifelong user modelling and personalisation in a web browser environment. We evaluated main characteristics of the proposed data structures. User interest tree enables us efficiently:

- to retrieve the most relevant terms (e.g., user interests, concepts, context)
- to retrieve the latest updated terms (e.g., temporal interests, fresh knowledge)
- to retrieve the relevance of given term (e.g., how much is it interesting for user, how well she grasps given knowledge concept)

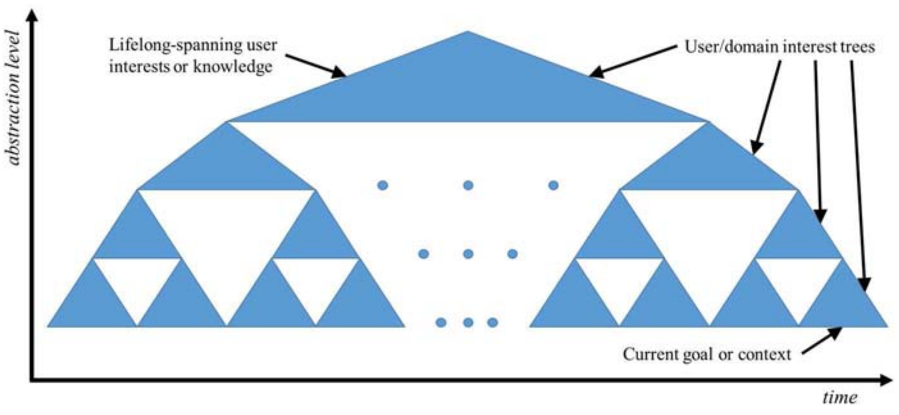
abstraction level
Lifelong-spanning user interests or knowledge
User/domain interest trees
Current goal or context
time

Fig. 2. Hierarchy of trees for different time and abstraction level.

- to retrieve the most relevant web pages from user web browsing history for a given term (e.g., which pages are interesting, contain particular concepts)
- to retrieve the least relevant or the oldest accessed term, which can be considered for removal (e.g., user has no more this kind of interest, has forgotten learned knowledge, changed context)

Domain interest tree broadens these possibilities by enabling us efficiently:

- to retrieve the most relevant terms within given domain (e.g., local interests)
- to retrieve the most relevant web pages from user web browsing history within a given domain (which pages are interesting within a given domain, are representative for a given concept)
- to retrieve the most relevant domains/subdomains (e.g., which domain is the most interesting one, contains the best grasped concepts

All these queries are just a top of the hill of possibilities. They represent only the most generic ones sufficient to cover most of common personalisation scenarios. There is much more of them depending on the chosen labels. All these queries (except those based on a simple term retrieval, which is linear to its length) can be done in time $O(k\times m\times\log_2(k\times m))$ as we already analysed above (k being number of topmost terms and m the length of the longest of them) using presented algorithm.

To wrap up this analysis and solve yet unanswered common question on magnitude of time complexity constant, we performed several experiments on a sample scenarios executed directly in the browser (see Table 1). We compare our indexer to various browser implementations of JavaScript object, which can be considered as the state-of-art implementations of commonly used associative array data structure. Note that operations like retrieval of the most relevant terms is not supported by default in JavaScript object. JavaScript object allows only one type of index to be used, which is the term itself in this case. To simulate retrieval of the most relevant terms, we need to copy all items into an additional array and sort it by relevance. Therefore, the time efficiency is the same regardless of how many items we retrieve. Such operations are rather demanding in various personalisation scenarios, since we want to recommend only the very relevant things (like movies, articles) and we do not need to bother with considering some less relevant interests (excluding local user interests, i.e. the most relevant terms within some particular domain, which are discussed in section about domain interest tree).

Table 1. Run time comparison of selected data structures in different browsers

Browser	Data structure	Term insertion	Retrieval by term	Retrieval of 10 most relevant terms	Retrieval of 50 most relevant terms
Chrome 17	Our indexer	1.77 μs	0.53 μs	200 μs	560 μs
Chrome 17	JS object	0.57 μs	0.04 μs	8650 μs	
IE 9	Our indexer	4.42 μs	3.54 μs	750 μs	1540 μs
IE 9	JS object	2.37 μs	1.4 μs	34610 μs	
Firefox 11	Our indexer	9.33 μs	8.06 μs	1090 μs	3410 μs
Firefox 11	JS object	1.7 μs	0.56 μs	13840 μs	

In our tests we used collection of 4 204 weighted terms (unique keywords potentially representing some reasonable interest) which were extracted by our BrUMo framework from web pages of a real user browsing history. All test results are given as an average of 100 test runs performed on Dell laptop with 2 GHz Intel Core 2 Duo under Windows 7 64-bit. In retrieval of the most relevant terms, our indexer clearly outperforms all browsers' implementations of JS object. Nonetheless, term insertion and term retrieval is still reasonably fast (in order of microseconds), which is sufficient for real-time usage in collaborative distributed lifelong personalisation. Interestingly, the ordering of browsers' performance differs between our indexer and JavaScript object.

5 Conclusions

In this paper, we presented an efficient representation of user characteristics suitable for limited web browser environment. We described a powerful labelling technique to index various aspects of user model. We proposed a method of lifelog indexing as well, which makes it a complete lifelong user modelling component. We demonstrated the real-world performance of the proposed indexer within BrUMo framework.

We focused on a widespread web browser environment, which implies computational limits and explains the importance of designing such low-level mechanisms for user model representation. Although straightforward in their principles, they are powerful enough to accomplish various recommender tasks and supports both collaborative and content-based filtering approaches commonly used in today's recommender systems. Since our experimental framework BrUMo enables communication among its multiple instances, users can be grouped together by comparing weighted vectors of their global interest (see section 3). Subsequently, similar users' models can be queried to retrieve the intended collaborative recommendations. As for content-based recommendation, this is even more straightforward since we already have user feature vectors. These can be retrieved by given webpage or web application to compare it to individual item feature vectors to compute the most relevant items. Since we are client-based, we avoid even the cold-start problem in both cases by sharing our private model (or parts of it) with multiple web systems.

Acknowledgements

This work was partially supported by the Scientific Grant Agency of Slovak Republic, grant No. VG1/0675/11 and by the Slovak Research and Development Agency under the contract No. APVV-0208-10.

References

1. Bilenko, M., Richardson, M.: Predictive client-side profiles for personalized advertising. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '11). ACM, New York, NY, USA, pp. 413–421 (2011).
2. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User profiles for personalized information access. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.): The Adaptive Web: Methods and Strategies of Web Personalization. LNCS 4321. Springer-Verlag, Berlin Heidelberg New York, pp. 54-89 (2007).
3. Gerber, S., Fry, M., Kay, J., Kummerfeld, B., Pink, G., Wasinger, R.: PersonisJ: mobile, client-side user modelling. In Proceedings of the 18th international conference on User Modeling, Adaptation, and Personalization (UMAP'10), de Bra, P., Kobsa, A., and Chin, D. (Eds.). Springer-Verlag, Berlin, Heidelberg, pp. 111–122 (2010).
4. Guo, H., Chen, J., Wu, W., Wang, W.: Personalization as a service: the architecture and a case study. In Proceedings of the first international workshop on Cloud data management (CloudDB '09). ACM, New York, NY, USA, pp. 1–8 (2009).
5. Hauger, D.: Using Asynchronous Client-Side User Monitoring to Enhance User Modeling in Adaptive E-Learning Systems. In: Dattolo, A., Tasso, C., Farzan, R., Kleanthous, S., Vallejo, D.B., Vassileva, J. (Eds.): CEUR Workshop Proceedings, Workshop on Adaptation and Personalization for Web 2.0 (UMAP'09). Vol. 485, pp.50-59 (2009).
6. Holub, M., Bieliková, M.: An Inquiry into the Utilization of Behavior of Users in Personalized Web. In: Journal of Universal Computer Science, Vol. 17, No. 13, 1830-1853 (2011).
7. Kim, K., Sho, S., Lim, J., Kim, S., Lee, S., Lee, J.: A client profile framework for providing adapted contents in ubiquitous environments. In Proceedings of the 5th international conference on Pervasive services (ICPS '08). ACM, New York, NY, USA, pp. 181–184 (2008).
8. Koidl, K., Conlan, O., Wei, L., Saxton, A. M.: Non-invasive Browser Based User Modeling Towards Semantically Enhanced Personlization of the Open Web. 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications. pp. 35–40 (2011).
9. Kramár, T., Barla, M., Bieliková, M.: Disambiguating Search by Leveraging a Social Context Based on the Stream of User's Activity. In: User Modeling, Adaptation and Personalization (UMAP'10). LNCS 6075, Springer, pp. 387–392 (2010).

10. Kramár, T., Barla, M., Bieliková, M.: Personalizing Search Using Socially Enhanced Interest Model Built from the Stream of User's Activity. In Journal of Web Engineering, Vol.12, No.1&2, pp. 65–92 (2013).
11. Matthijs, N., Radlinski, F.: Personalizing web search using long term browsing history. In WSDM '11: Proc. of the fourth ACM international conference on Web search and data mining, ACM Press, pp. 25–34 (2011).
12. Mihalcea, R., Tarau, P.: TextRank: Bringing order into texts. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 404–411 (2004).
13. Mika, P.: Ontologies are us: A unified model of social networks and semantics. Journal of Web Semantics Vol. 5, No. 1, pp. 5–15 (2007).
14. Ohmura, H., Kitasuka, T., Aritsugi, M.: A Web Browsing Behavior Recording System. In Lecture Notes in Computer Science: Knowledge-Based and Intelligent Information and Engineering Systems, Springer, Berlin, Vol. 6884, pp. 53–62 (2011).
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management, Vol. 24, No. 5, pp. 513–523 (1988).
16. Velayathan, G., Yamada, S.: Behavior based web page evaluation. Journal of Web Engineering Vol. 6, No. 3, pp. 222–243 (2007).