THE UNIVERSITY OF SYDNEY

SCHOOL OF INFORMATION TECHNOLOGIES

MNEME: A FRAMEWORK TO SUPPORT PEOPLE IN THEIR SISYPHEAN TASKS

TECHNICAL REPORT 663

JUDY KAY AND BOB KUMMERFELD

DECEMBER 2010

Mneme: a framework to support people in their Sisyphean Tasks

Judy Kay and Bob Kummerfeld

School of Information Technologies, University of Sydney, Sydney, NSW 2006, Australia
{judy,bob}@it.usyd.edu.au

Abstract. This paper introduces the notion of *Sisyphean Tasks*, ones that people need to do repeatedly, for long periods, possibly over many years, to achieve important goals. Some examples are associated with maintaining or improving health and fitness: doing regular exercise; tracking one's weight; taking medication properly. Such tasks have driven much pervasive computing research precisely because it has the potential to easily capture relevant information, track a person's performance and activity, and make effective use of that information. We identify the design requirements for a user controlled infrastructure to support Sisyphean Tasks by supporting four important classes of applications: *mirrors*, *ambient displays*, *alerts* and arbitrary *personalized applications*. We describe our implemented architecture for this and report its evaluation based on creating demonstrator applications. Our key contributions are: defining the nature of Sisyphean Tasks; designing an infrastructure to support them, both enabling aggregation of information and supporting flexible tools that use this to support Sisyphean Tasks; creation of Mneme, our implementation of that design; and its validation via applications that have had long term use.

1 Introduction

Pervasive computing has the potential to provide valuable support for people to achieve long term goals in many critical aspects of their lives. This is because it offers the promise of harnessing data that is captured by a range of sensors, throughout the user's environment. One important class of these relate to the long term goal to maintain or improve health. Pervasive computing sensors can support this by making it easy to capture relevant data about factors affecting health, such as nutrition, sleep and exercise, as well as indicators of health, such as weight, rest pulse and blood pressure. For example, the UbiFit system [4] made it easy to for a person to track their activity and it then provided a glanceable display of this information. Over three months, UbiFit users were better able to maintain physical activity levels because of the automatic data collection on activity combined with presentation of activity summaries in an effective format on their mobile phone, a device that was readily available.

The goal of maintaining health, and its subgoal of achieving or maintaining healthy levels of activity, is one example of a very long term goal. We call these

Sisyphean tasks, ones that people are compelled to do repeatedly, as part of achieving their important goals. Like Sisyphus, we need to continue to exercise if we want to be healthy. There are several important classes of Sisyphean tasks. Some that have had considerable attention in the pervasive literature relate to health, for example, aiding adults to manage diabetes [23] and a design to provide similar support for children [26]. Another class of Sisyphean tasks relates to long term learning and skill acquisition, for example, tracking a child's long term development [16].

There are two key ways that pervasive computing can help a person tackle their own Sisyphean tasks: making it easy to capture relevant data; and to exploit that data making it available pervasively when and where it is useful for supporting the tasks. Importantly, there are cases where pervasive computing applications could help the individual with their Sisyphean tasks. For example, many people must take medication at particular times daily, an important Sisyphean task for them. Pervasive applications can help with this in several ways. They can make it easier to record when the medication has been taken. They can then support a mobile interface for checking if it has been taken. This can give peace of mind if the user goes out and cannot recall whether they took their medication. The mobile phone can also issue alerts when the medication is overdue. An ambient display can help remind a persons if they have not taken their upcoming dose. In case of people with memory loss, such systems can also be valuable for their relatives and carers [5].

We are beginning to see the emergence of many devices that can automatically capture valuable information for Sisyphean tasks. For example Withings scales[1] wirelessly transmit their readings for a person's weight and fat percentage to the Withings server. Other tools can wirelessly capture and record activity levels[2]. These share some aspects of the philosophy of lifelogging [10] which aims to automatically collect rich sets of information about people, such as video streams. Another class of "sensor" can make use of personal information management (PIM) tools such as a todo-list manager[3] which enables a person to define tasks such as walking for 30 minutes each day. If the user then marks each of these tasks as done, or not, this creates a valuable source of data about activity. Another recently emerging class of "sensor" supports Personal Informatics [22]; they help people collect information about themselves conveniently.

At present, many of these sensors store their data about the user in various *silos*. This means that a person needs to consult several different programs and web sites to see their information. Nor is it easy for the individual user to make various parts of the data available to arbitrary pervasive computing applications that could aid them in tackling their Sisyphean tasks, such as the elements we described above for helping a person with achieving good compliance in taking medication.

[1] device www.withings.com/

[2] http://www.directlife.philips.com/, http://www.fitbit.com/

[3] like Remember the Milk https://www.rememberthemilk.com

We designed the Mneme framework to support people in their Sisyphean tasks. It can capture a broad range of relevant sensor information, both from automatic capture of data by pervasive sensors and from various interfaces that enable the user to record data. It can combine the information in flexible ways and can make it available to the range of applications that can play a role in supporting Sisyphean tasks. Importantly, from the very foundations of its design, it aims to put the user in control of this whole personal infrastructure. In the next section, we review related work. Then we analyze the nature of Sisyphean tasks and explain how this influenced the high level design of Mneme. We then present its architecture and implementation, followed by our validation, based upon creation of a set of demonstrator applications. We conclude with a discussion of the implications of our approach and future work.

2 Related Work

The purely technical aspects of our work build upon several areas of research that aim to exploit emerging technologies that enable people to capture, retain and use long term collections of digital information about their lives [8]. All have a common foundation: it is technically feasible to collect huge amounts of data about ourselves; and that information has the potential to be of great value in the long term, perhaps for purposes that were not envisaged when it was collected.

At one end of the spectrum is work that aims to automatically capture rich multimedia of a person's daily activity, including video and audio records of the day as in MyLifeBits [10]. This approach is characterized by its use of raw data, with automated capture, and no need for explicit user action to define the salient parts [24]. In one study, involving people with severe cognitive loss [21], it showed promise in aiding memory formation and recollection when used to review a day's lifelogs. A study of lighter forms of lifelogging on phones [15] aided social goals and highlighted the need for user control over the automatically logged personal information. Research challenges associated with lifelogging [8], include determining what is important, aggregating information from multiple sources, ensuring safe storage and management and effective sharing. A recent critique of lifelogging [25] points to the need to build on previous work, particularly that which points to potential benefits to memory. Our work does just this.

Others have explored ways to aggregate a person's information, rather than leaving silos of it across various web sites. For example, SUMI [20] does this by creating a personal portal for social networking information, aggregated from several sites. It tackled the difficult problems of different ontologies across social network sites as well as the interface challenges associated with supporting user control over their own information.

Our work builds from the goals of the many pervasive computing researchers, who have explored ways to support health needs. For example, [23] automatically captured evidence of activity as well as manually entered data such as blood sugar and blood pressure readings, presenting these on a phone. This is similar to

[4] which automatically captured activity and provided phone-based summaries. Unlike these individual systems, our work aims to give people the ability to make use of any information sources that suit their needs and budget, aggregating all their own information so that it can then drive applications of their choice.

This also differentiates it from the emerging range of commercial systems in this space, such as Google Health and Microsoft's HealthVault, which enable a person to aggregate of a range of information from their set of sources and medical providers. One view of all these is that each of these constitute yet another web silo of the user's personal information; however, both aim to be repositories that also aggregate data from health providers. Our work takes account of the notion of *personal space extension* [6] describing the trend for people to put personal information online for flexible access.

None of the previous work has explored ways to give a user the means to easily create and control their own infrastructure, defining the aspects of themselves that they wish to model and establishing the evidence sources that can contribute to each part of the model as well as defining which applications can make use of the model. This is the goal of our work.

3 Analysis of Sisyphean tasks and design for mental model

We now describe our analysis of Sisyphean tasks to inform the design of an infrastructure to support them. First, we identify the attributes Sisyphean tasks[4]:

- they must be done *repeatedly*;
- over the *long term*;
- to achieve *important long term goals*;
- for *improvement* or *maintenance* of an outcome;
- pervasive computing can provide a sources of evidence about *success* in meeting the Sisyphean goals or *effort* in completing the Sisyphean tasks towards those goals;
- they have to fit into people's lives over the long term and so supporting them must take into account that people have limited *attention*, should not *interrupted* unnecessarily but should be interrupted when that is justified.

To aid our description of our design process, we refer to Figure 1 which summarizes it in terms of the mental model our work aims to support. In the center is Mneme[5], so named because it is intended to *remember* useful information. Essentially, it is a repository for all the information to support Sisyphean tasks.

Goals driving Sisyphean tasks

Consider the nature of long term goals that motivate Sisyphean tasks, such as the following which have motivated significant pervasive computing research:

[4] We have italicized the key concepts that were critical for the design.

[5] In Greek mythology, she was the muse of memory

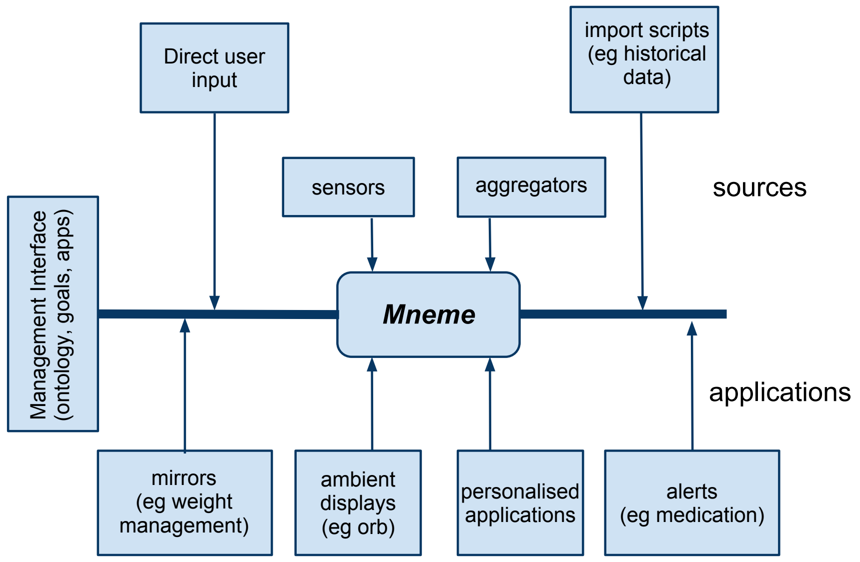

Direct user input
import scripts (eg historical data)
sensors
aggregators
sources
Management Interface (ontology, goals, apps)
Mneme
applications
mirrors (eg weight management)
ambient displays (eg orb)
personalised applications
alerts (eg medication)

Fig. 1. Overview of Mneme vision to meet design goals to enable people to:
G1. Establish goals motivating Sisyphean tasks.
G2.1 Link these to *sensor* data reflecting effort and success.
G2.2 Easily input *explicit* information about the effort, success or context.
G2.3 Easily aggregate arbitrary web-based information.
G2.4 Easily incorporate historic information.
G3.1 See *mirrors* of useful information about progress.
G3.2 Establish *ambient displays* to support Sisyphean tasks.
G3.3 Establish *alerts* to support Sisyphean tasks.
G3.4 Link *personalized applications*.

– *physical goals* eg healthy weight, blood pressure, activity levels [4], following required medication regimes [5], tracking information for managing diabetes [23], posture [12], self care for the elderly, [5], performance in sport [2];
– *mental, spiritual, emotional and social goals* eg fulfilling responsibilities to immediate family, maintaining broader social links, positive mood;
– *intellectual and creative goals* eg learning a new language, learning about fitness and health, long term intellectual development [16], craft projects, achieving expertise in an area;
– *environmental goals* eg reduce electricity consumption [18] reduce water use [9], improve air quality [17], improve driving behaviors, make more use of public transport and walking rather than driving;

This is not intended to be exhaustive, but rather to characterize the breadth and nature of the goals. Even with this indicative list, it is clear that there are many potential goals that could involve Sisyphean tasks. It is also apparent that our design must take into account that people have limited attention. So it would be infeasible to keep all of these at the forefront of one's attention. Moreover, different people will set different priorities. So, while some goals, such as those associated with physical well being, are likely to be important to many people. others will be particular to individuals. Over time, one person's priorities will change. For example, a person who learns that they have diabetes may need to set new goals or increase the priority on existing ones. This leads to our design goal for our infrastructure, to enable person to *establish their own goals motivating their Sisyphean tasks (G1)*. So we need a management interface that enables the user to define such goals as shown at the left of Figure 1.

Capturing all relevant information about effort, success and context in Sisyphean tasks

We now consider the range of sources of information that could provide valuable information about Sisyphean tasks and the achievement of long term goals. At the upper left of Figure 1, we show two key sources of information about Sisyphean goals.

Sensors include all mechanisms that can capture information without requiring effort from the user. New ones are becoming available. For example, Withings scales[6] wirelessly transmit their readings for a person's weight and fat percentage to a Withings server. Other tools can wirelessly capture and record activity levels[7]. Our goal is to enable people to make use of any sensors that they wish, flexibly linking these to any of their goals. So, for example, an activity sensor could inform the goal to loose weight, to improve mood, and to maintain posture because it indicates that the user spends less time sitting. This leads to our design goal that the infrastructure should enable the user to *link arbitrary sensors with any of their long term goals (G2.1)*.

While such sources are important, it is also critical that people can give *direct input* of any information they wish. This may partly offset the limitations of the existing technology. For example, in one study involving automatic activity tracking [4] people expressed dissatisfaction that it failed to capture some activities. Had they been able to easily record this, it would partly address this limitation. In general, it is desirable to ensure that people can flexibly capture any information that they wish to. This motivates the design goal that the infrastructure should support users to *directly input information (G2.2)* where this may include a record of their *effort* such as an exercise session, their *success*, such as beating a personal best, or *context* such as a note about a landmark event that might help explain changes or support later retrieval of information.

Our infrastructure should enable a person to exploit relevant personal information that is currently managed by diverse applications and within web

[6] device www.withings.com/

[7] http://www.directlife.philips.com/, http://www.fitbit.com/

services. This includes emerging health information portals[8]. Depending upon the health services one uses, useful information may be available from one or more sites like these. Many personal information management (PIM) tools also hold information that could be useful. For example, a todo-list manager [9] enables a person to use their mobile phone or desktop to plan, track and recall things that they want to do. To exploit these, a design goal is to enable the user to *aggregate arbitrary web-based information, linking it to their Sisyphean tasks (G2.3).*

The fourth source of information is any data the user had captured in the past. For example, the user may have kept a personal log of their weight, a spreadsheet of the long term sports training information or the like. At the point that the user realizes that this might usefully contribute to the long term picture of their progress on Sisyphean tasks, they should be able to incorporate it. This gives the design goal that users should be able to *easily incorporate historic information (G2.4)*

Exploiting information about Sisyphean tasks to help achieve long term goals

How should be make use of information about Sisyphean tasks? A recent critique of lifelogging [25] suggests some important possibilities. It notes the potential benefits of helping people recollecting, reminiscing, *retrieving*, *reflecting*, and *remembering* intentions, where the italicized elements are of particular value for supporting Sisyphean tasks if there is a suitable interface that the user can access when they wish to do one of these actions.

We consider first, support for retrieving relevant information to support reflection. This can take the form of *mirror* applications which present information in a form that helps users *see* themselves, supporting awareness, reflection and planning. This has been shown to be valuable in a broad range of contexts. For example, Open Learning Modelling (OLM) systems [3] have been valuable in improving learning by enabling the learner to become aware of their level of performance, track their progress and plan their learning. In the health context, the UbiFit glanceable display of user's physical activity [4] enabled people to maintain a higher level of physical activity. In long term work by small teams, visualizations of models of each team member's activity helped identify group problems, supporting monitoring of progress and had an additional role as a new means to navigate large collections of digital materials [27]. Currently, many applications and web sites provide helpful interfaces to the particular data that they manage, for example in association with the Fitbit[10], Garmin FR60 pulse recording watch[11] and Withings wireless bathroom scales[12]. We aim to make all

[8] such as Google Health https://www.google.com/health, Microsoft's HealthVault http://www.healthvault.com/
[9] such as Remember the Milk https://www.rememberthemilk.com
[10] fitbit.com
[11] http://www.garmin.com/
[12] withings.com

such information available in one place, so the user can reflect on any combination of the information they wish. This gives our goal to support users to *access mirrors showing information associated with Sisyphean tasks (G3.1).*

Taking account of people's limited attention, the need to remember intentions suggests that we should also need an active system, that can enable a user to delegate the job of remembering their Sisyphean tasks. We distinguish these in terms of whether they *interrupt* the user. We aim to enable the user to establish *ambient displays that present useful information about progress (G3.2).* These do not interrupt the user.

There are cases where interruption is desirable. For example, if the user wants help in remembering to take their medication, they may want to be interrupted once a dose is overdue. This motives our design goals to enable the user to establish *alerts to support Sisyphean tasks (G3.3).*

The final element in our design is to *support for arbitrary personalized applications (G3.4).* For example, an exercise coaching system may personalize its advice based on rich collections of information about the user's past activity and their response to previous advice.

3.1 Control interface

For all the goals, an essential design goal is that the user be able to control each element, from the definition of goals, linking these to information sources, be that via direct interfaces, sensors, aggregators or import of historic data and establishing which applications that make use of the relevant information to provide mirrors, ambient displays as well as active notifications. This is shown at the left of Figure 1.

4 Architecture and implementation of Mneme

We now describe the architecture of our infrastructure to meet the design goals just described. We illustrate it in Figure 2. The oval indicates the two parts of the Mneme framework. At the top is the Mneme portal which manages interaction with the user. At the upper left of the figure, we show the functionality it provides for managing and viewing the Mneme models. At the upper right, we show its facility for direct user input. The other part is the Personis [1] user modelling framework. The remainder of the architecture is based on applications, each of which interacts with Mneme via the *app* shown in the figure.

4.1 Mneme information store

This is the backend representation and storage for Mneme. To implement this, to chose the PersonisAD modelling system [1] as a foundation, because it supports the representation of a hierarchical namespaces called *categories* which contain *components*, each representing one key aspect, where this can be an *attribute* such as the user's weight, a *preference* such as preferred foods or exercises, *knowledge*

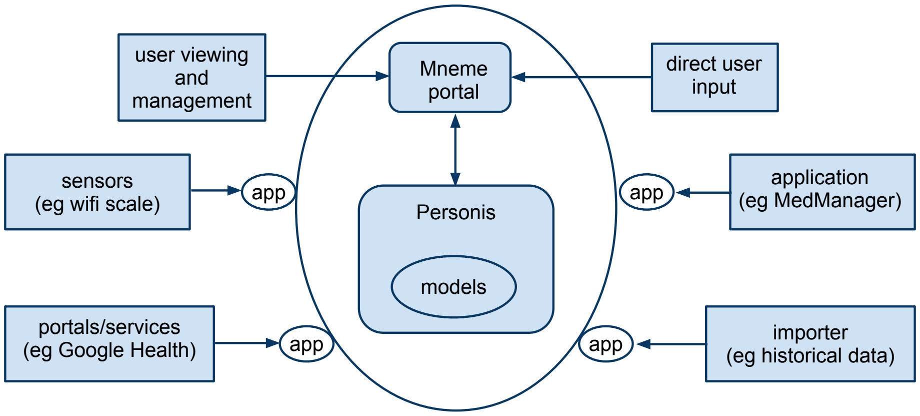

user viewing and management
Mneme portal
direct user input
sensors (eg wifi scale)
app
Personis
app
application (eg MedManager)
models
portals/services (eg Google Health)
app
app
importer (eg historical data)

Fig. 2. Mneme architecture.

(for supporting long term learning). It also has flexible mechanisms for reasoning, privacy control and was designed to support explanations of its operation as a basis for user control [7]. PersonisAD is *active* in the sense that components can have rules which fire when new information is added to a component. This can be used to add information to other parts of the model or to drive an external application. It is *distributed* in that parts of the model can reside on arbitrary machines. It is based on the *accretion/resolution* representation. This allows information sources to *tell evidence* about components; it *accretes* this, without interpreting it. It time-stamps such evidence and tags it with its source (such as the name of sensor). When an application *asks* for information about components, PersonisAD *filters* evidence according the rights of application, for example only allowing information from some sources to an application. Then it uses a *resolver*, one allowed for that application, to determine how to interpret the available evidence. This flexible system supports reasoning based on noisy and uncertain evidence. For Mneme we needed to enhance this representation. We now describe the new features needed for Mneme.

One requirement for Mneme, was to represent goals. To do this, we augmented Personis so that a category or component can be defined to be a *goal*. It can then represent the following additional parts.

– *Target*. This is optional but can specify a value that is the user's target. This is added to the component's evidence list, timestamped and typed as a target value. Over time, a user may nominate differing targets and these are all kept.
– *Child components*. This is an optional list of components the user wants to associate with this goal. These can be subgoals (other goal components). For example, if a user defines the goal called *health*, they may link this to

components, goal components, such as *Exercise sessions per week*, *weight* or not, as in the case of *Blood Pressure* or *Health Note Log*.
– *Parent goal list.* This is a backward link to goal components or categories for which this is a child. One component may be part of several goals. For example, if the user thinks of *walking to work* as contributing to *health* and *environmental goals*.

The other major facilities added to Personis were to support the flexible addition of applications. First we needed to support definitions of links between arbitrary parts of the Personis model. This enables the user to link a goal with evidence that is collected by application. So, for example, the user can link their *weight* component with the evidence from the *Withings application* which makes use of weight readings on Withings bathroom scales. We explain these in the next subsection.

We also needed mechanisms to import and export *partial models*. The import is critical for the task of adding new information sources and applications. This allows new areas to be modelled conveniently: an application can initialize a section of the model by importing a partial model template that contains only the component ontology and no information about the components. The export is needed for applications that do rich personalization, requiring a many components below a point in the hierarchy. For example, this may be useful for applications that teach the user and need detailed models of their knowledge in a domain. Existing Personis mechanisms for filtering are used for managing privacy. We also use both import and export for maintenance and creating backups. We envisage they would be useful for exporting part of the model to an external device such as a phone so that it can operate when disconnected [11].

4.2 Mneme *apps*

A key new element was required to support management of arbitrary applications that either provide information to the model (via the *tell* primitive operation) or make use of it (via the *ask*). The new facilities support registration of new applications. An application is registered by the owner of the model and a security key generated. The application then uses the key to carry out operations on the model. Permission for the application to perform ask/tell operations is controlled for each category in the model, with the default being no permission.

The framework requires an *app* to be registered with Mneme, enabling a user to associate them with parts of their model. Each *app* communicates with the Mneme models using Personis *ask* and *tell* functions. The simplest *app*s operate as glue code between an external system and the model. More sophisticated *app*s may have substantial functionality of their own, as we illustrate in the next section. Figure 2 shows the four examples of classes of *app*s: sensors which operate in real time to *tell* information to the model as in the case of wirelessly connected scales which send weight and fat measures to an application provided by the manufacturer. This connects to a small *app* that we created to *tell* this information to Mneme. The figure shows the class of *app* that interacts with one of the many online portals that have APIs, such as Google Health, to *tell* information to Mneme. We show the class of *app*s that users choose to explicitly invoke to import information from their desktop, *tell*ing this to Mneme. This serves as a bridge between Mneme and data from arbitrary files and exported from applications. The final class of *app* is a more general application which does both *tell*s and *ask*s on the models. The next section illustrates all four classes of *app*s.

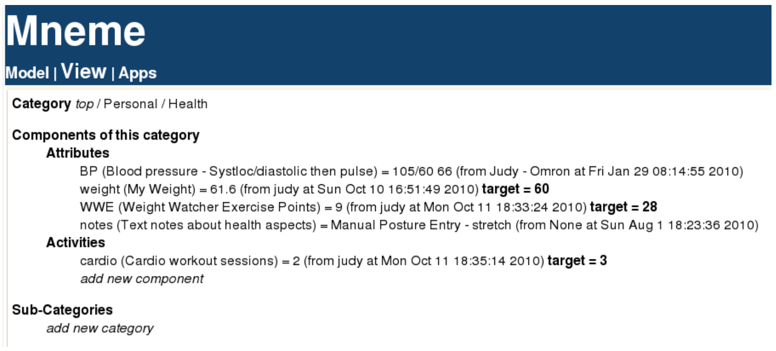
A key goal for our design of the architecture and implementation is to minimize the effort for programmers to create new *apps* that enable the user get information into their model, as well as *apps* that make use of the model to personalize their actions. All current applications (Withings, graph-weight, MedManager etc) are written in Python. In the next section, we describe several *apps* we have created. They range in code size from 22 lines to 500 lines of Python. The "Withings" *app*, at 22 lines, is a script that is invoked from the withings.com site when a new piece of weight data is available. The script fetches the weight data and performs a "tell" operation to add it to Mneme. The MedManager is approximately 500 lines, driving a complex system that delivers multiple alerts and ambient displays.

4.3 Mneme portal

Mneme provides a web-based interface which enables the user to manage their model. Figure 3 shows the user model *View* of the *Personal/Health* category (some detail has been ommitted for clarity). This has 6 components. The first, *BP* maintains evidence about blood pressure. The screen shows the reading from Jan 9, 2010 and shows that the source was the user (Judy) using the Omron device. Next is the user's current weight goal, which currently has a target value of 60 and based on the last reading (Oct 10, 2010) its current resolved value is 61.6. For any component, clicking its name causes an expanded display showing the recent evidence and the ability for the user to add new evidence for this component. This was how the user entered the blood pressure, WWE and notes shown in this page. The other facilities on this page are the clickable *add new component* for creating new components and *add new category* for creating a subcategory within this one.

Within the *Apps* page, example shown in Figure 4, the user may add a new application. This enables the user to select and configure the applications that they wish to be used in conjunction with their Mneme user model. An example configuration page for the *Fitbit* application and associated device [13] is shown in Figure 5. The *Fitbit* application operates in conjunction with a body-worn device that sends readings for number of steps taken each day to the Fitbit service. The Fitbit service can be configured to send this information, along with an estimate of the number of kilometres walked and calories expended, via a twitter message. The *Fitbit* application parses the twitter message and adds the information to the user model.

[13] http://fitbit.com



Category *top* / Personal / Health

Components of this category

Attributes

BP (Blood pressure - Systloc/diastolic then pulse) = 105/60 66 (from Judy - Omron at Fri Jan 29 08:14:55 2010)
weight (My Weight) = 61.6 (from judy at Sun Oct 10 16:51:49 2010) **target = 60**
WWE (Weight Watcher Exercise Points) = 9 (from judy at Mon Oct 11 18:33:24 2010) **target = 28**
notes (Text notes about health aspects) = Manual Posture Entry - stretch (from None at Sun Aug 1 18:23:36 2010)

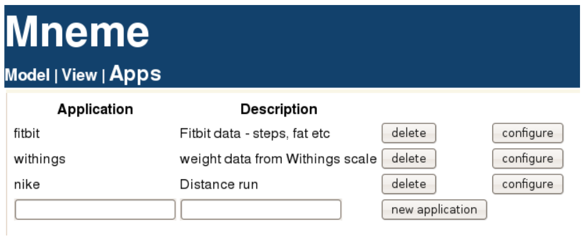
Activities

cardio (Cardio workout sessions) = 2 (from judy at Mon Oct 11 18:35:14 2010) **target = 3**
add new component

Sub-Categories

add new category

Fig. 3. Example of Mneme screen that gives *View* of the model and enables the user to create new parts of the model and add evidence to components.



Application	Description		
fitbit	Fitbit data - steps, fat etc	delete	configure
withings	weight data from Withings scale	delete	configure
nike	Distance run	delete	configure
		new application	

Fig. 4. Applications selection page.

The *configure* button on the *Apps* page allows the user to link the relevant components in their model with those in the application. So, for example, if the *Withings* application uses a term other than the user's chosen name for the *weight* component, the user maps these in the configuration screen.

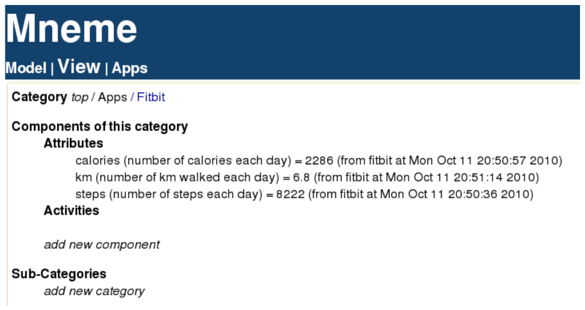
In summary, Mneme provides interfaces that enable the user to define new categories and components in their model, add evidence and activate applications that can contribute to or make use of the model.

The Mneme portal is implemented as a web application using a combination of HTML/CSS, Javascript and Python. Basic page layout uses the Cellerator web framework [14]. Mneme runs under Cellerator on a web server as a cgi script. Communication between Mneme and the Personis server uses an RPC mechanism with http message transport and JSON message encoding.

5 Demonstrator applications

As one of our core contributions is the creation of the new framework, Mneme, we evaluated it by demonstrating its use to create two demonstrator applications, as recommended in [14]. The first of these is a minimal example of a mirror,

[14] http://www.cs.usyd.edu.au/~piers/



Category *top* / Apps / Fitbit

Components of this category

Attributes

calories (number of calories each day) = 2286 (from fitbit at Mon Oct 11 20:50:57 2010)
km (number of km walked each day) = 6.8 (from fitbit at Mon Oct 11 20:51:14 2010)
steps (number of steps each day) = 8222 (from fitbit at Mon Oct 11 20:50:36 2010)

Activities

add new component

Sub-Categories

add new category

Fig. 5. Fitbit information.

to support reflection on long term data. The second is more complex, involving several small *apps*.

5.1 Mirror application examples

An important goal for many people is to improve or maintain their fitness and health. This involves a complex set of behaviors, such as exercising regularly and making healthy eating choices. There are several relevant measures indicating progress and success, including weight, fat levels, blood pressure and rest pulse. There are already many web sites which enable a person to monitor such information associated with this goal [15], reflecting its importance for many people. We have also mentioned the emerging set of tools that *automatically* capture relevant information, and wirelessly transmit it to a central site. Then, users can view this data at the associated web site. Essentially, each of these provides its own mirror interface, but they are all independent and different. We now show how Mneme can aggregate such sources. We also give an example of a mirror applications which enables the user to see this information.

Using the interfaces described in the last section, a user can define a Mneme health context with relevant components, such as *weight*, *fat level*, *blood pressure*, *exercise levels*. Then they register the relevant applications (as described above). For this validation, applications which use the API available for a site to add to the user model were:

- *Withings* which makes user of the API provided by Withings, for their wireless scales for weight and fat levels, to add these to the *weight* and *fat level* components in the Mneme model;
- *Garmin* for pulse rate readings for the *rest pulse* component;
- *RTM* which makes use of the task manager system, Remember The Milk [16], extracting just the tasks that the user titled *daily exercise* and adding this to the exercise levels component.

[15] such as http://www.weightwatchers.com, http://www.coolrunning.com/
[16] https://www.rememberthemilk.com

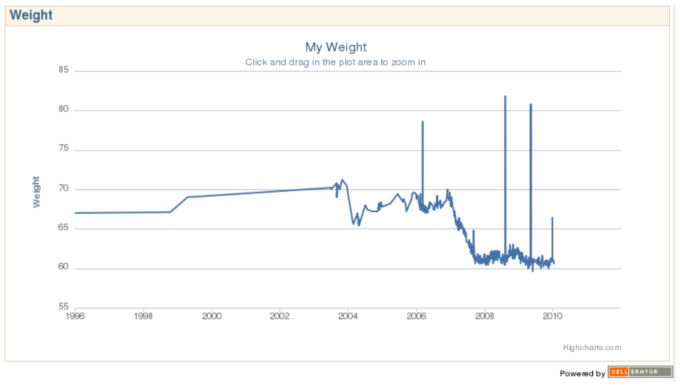


Fig. 6. Example mirror interface.

We used a set of *importer* applications to incorporate data that the user had captured over several years, using custom desktop applications. This provided evidence for the following components:

– *weight* from data the user had captured from 1996 until October 2010;
– *exercise levels* from 277 records of exercise points (where an hour's gentle exercise is about 3 points) from February 2007 until November 2009 as well as 213 pedometer readings from January 2006 until July 2007;
– *BPP* which added *Blood pressure* and *rest pulse* data.

Finally, for a period of one month, the user directly entered readings into Mneme for *Blood pressure* and *exercise levels* information.

This set of applications are representative of each of the four classes of user model evidence sources in Figure 1. Once these processes have populated the Mneme model, the user registered applications that provide the *mirror* interfaces. Figure 6 shows output from one of these, with the user's model for weight. Data from 1996 until November 2009 was from old data, mined by the *datalist* application and subsequent data came from the *Withings* application. There are some clearly anomalies in this data, the first due to errors in data entry by the user and the last due to the Withings scale incorrectly associating another person's weight reading with them. In previous work, we have used resolver functions to address problems like this [1].

5.2 Pervasive demonstrator

There is a large, and growing, group of people who need to take life-saving medications over the very long term. This poses challenges, as reflected in a recent study [19] which observed that people, especially the elderly find it difficult to maintain their medication regime, "forgetting to take their medicine, uncertain if they have taken their medicine before, taking extra dose of medicine when they have already taken, taking the wrong types or amount of medicine, and running short of medication supply unexpectedly". While some very simple packaging [13] can help a person to check whether they took their medication by looking at the package, this is not useful if the user is away from the location of the medicine. The MedManager application illustrates how Mneme has been used to help a user with some of these problems, by providing:

– a web interface and phone application where the user can check whether they have taken their last dose;
– a mobile phone alert sent if the user failed to take their medicine in time;
– an RFID-based *invisible* application which simplifies tracking when the user has taken medication.
– an ambient display, based on an orb, which glows red when the user's medication is overdue, green otherwise;

All make use of a component in the model called *taken* [17] where MedManager adds evidence reflecting each time the user has taken medication.

We now describe the operation of the web interface and phone application. At either the web of phone interfaces, the user can click a button to indicate taking medication. This performs a *tell* on the *taken* component. (Of course, for the system to be effective, the user needs to remember to do this when they take the medication and this makes the phone application the more convenient and effective.) When the web and phone application starts, it performs an *ask* on the *taken* component and displays the last two times medication was taken.

Now consider the way MedManager sends a reminder when the user has failed to take their medication. This has several parts. A *timer* within MedManager has the times that medication should be taken and at those times, it *ask*s the *taken* component for the most recent record of the user taking their medication. In our operating version, the user takes two sets of medication, one for 8am and one for 8pm. If the *timer* determines the user has failed to take their medication, it activates the *reminder* and begins sending alerts to the user's mobile phone every 15 minutes until the *taken* component indicates it has been taken.

Even finding a phone at the time of taking medication can be inconvenient and may mean the user fails to do it. So MedManager has an RFID-based mechanism designed to make it almost no extra effort to tell MedManager when medication has been taken. Figure 7 shows a cabinet in the user's living room. The leftmost picture shows two containers, one for each of morning and evening medications, the next picture shows the contents of one of the containers. The following picture shows the RFID reader, installed inside another attractive box and the RFID tag on the back of one of the medication containers. Finally, the rightmost image shows the user passing a container close to the RFID reader causing MedManager to do a *tell* on the *taken* component. MedManager also produces a spoken audio message confirming that the user's action has been recognized by the system.

The Orb light, also shown in the photos, is normally a dim green color but changes to a pulsing red if the medication is overdue. In addition, a soft tweet-like tone is played every few minutes if medication is overdue.

[17] within the user's Mneme *personal/health/medications* category

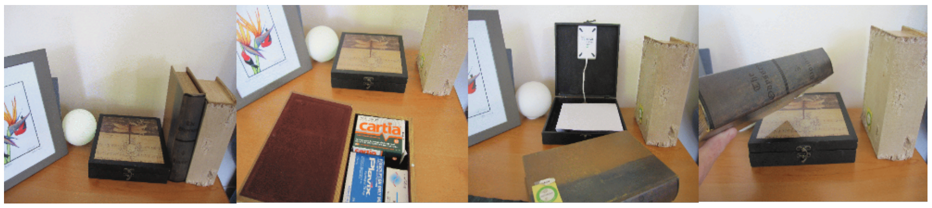


Fig. 7. RFID-based part of MedManager.

6 Discussion and conclusions

Our goals were to design and implement an infrastructure to support people in their Sisyphean tasks, enabling them to *control* their own long term personal information, aggregating personal data from a range of services and systems. This goal meant we needed mechanisms for the user to define their goals, with terms that they choose. In Mneme, users do this by creating categories to structure their user model and components which within these. Another key part of Mneme enables users to determine the applications that contribute information about either their activities or measures of their success in their Sisyphean tasks.

Our work complements the emerging cloud computing services with their convenient, any time, any place access to personal information. These services tend to *fragment* a person's information, exacerbating the long standing issues of fragmentation across the user's applications and across their computers and other devices. Mneme addresses this problem. Importantly, Mneme gives the user the convenience of cloud services without sacrificing control over that data.

We have identified four classes of applications that should contribute information about Sisyphean tasks: automatic sensors, aggregators, direct user input and the harvesting of historic data. We have identified three classes of applications that can help the user exploit this information in achieving Sisyphean tasks: *mirrors* enable people to *see* aspects of themselves more clearly, but operate only when the user decides to activate them; ambient displays that operate automatically but subtly, displaying information; active notifications; and arbitrary personalised, potentially pervasive applications that can make more complex uses of the Mneme store.

Mneme provides a framework for programmers to easily create such applications. Then the Mneme portal enables the user to choose which of these applications may contribute to or use particular parts of their model. Mneme currently provides a framework and useful demonstrator applications. We are building additional applications for educational roles for the lifelong learning. To date, the actual use of Mneme has been restricted to the authors over 15 months, and incorporating data over 14 years. User trials are an important next stage; these will provide a quite different understanding and privacy limitations will make this far more restricted. Future work include evaluating user interfaces, exploring how readily users choose to use Mneme over the long term. We want to add mechanisms to enable users to determine the storage of different parts of the model on different machines, to match their privacy concerns. We will also need to assess ease of *app* creation by programmers and avoiding security problems.

We have demonstrated that the Mneme framework can integrate evidence from four important source classes: sensors; cloud services; direct user input and importer sources which can mine the user's personal information stores. We have validated the framework with two useful demonstrator applications, one for mirroring health and fitness data and the other supporting a pervasive computing application that helps a person remember to take medication, based on alerts and ambient displays. This work makes a contribution as the first infrastructure designed to enable an individual user to create and manage the set of applications and management of their information stores as a basis for controlling the use of this information to support Sisyphean tasks. And a key contribution is the analysis of the nature of Sisyphean tasks and the ways that pervasive computing can support them by making use of the Mneme infrastructure.

Acknowledgments Removed for blind review.

References

1. M Assad, D J. Carmichael, J Kay, and B Kummerfeld. PersonisAD: Distributed, active, scrutable model framework for context-aware services. In *Pervasive 07*, pages 55–72. Springer, 2007.
2. Marc Bächlin, Kilian Förster, and Gerhard Tröster. Swimmaster: a wearable assistant for swimmer. In *Ubicomp '09: Proceedings of the 11th international conference on Ubiquitous computing*, pages 215–224. ACM, 2009.
3. S. Bull and J. Kay. Student Models that Invite the Learner In: The SMILI:() Open Learner Modelling Framework. *IJAIED*, 17(2):89–120, 2007.
4. S. Consolvo, P. Klasnja, D.W. McDonald, D. Avrahami, J. Froehlich, L. LeGrand, R. Libby, K. Mosher, and J.A. Landay. Flowers or a robot army?: encouraging awareness & activity with personal, mobile displays. In *Ubicomp*, pages 54–63, 2008.
5. Sunny Consolvo, Peter Roessler, and Brett E. Shelton. The CareNet display: Lessons learned from an in home evaluation of an ambient display. In *UbiComp 2004: Ubiquitous Computing*, pages 1–17. Springer Berlin / Heidelberg, 2004.
6. Y Cui and V Roto. How people use the web on mobile devices. In *WWW '08*, pages 905–914. ACM, 2008.
7. Marek Czarkowski and Judy Kay. *Giving learners a real sense of control over adaptivity, even if they are not quite ready for it yet*, pages 93–125. IDEA, 2006.
8. M. Czerwinski, D. W. Gage, J. Gemmell, C. C. Marshall, M. A. Pérez-Qui nones, M. M. Skeels, and T. Catarci. Digital memories in an era of ubiquitous computing and abundant storage. *Commun. ACM*, 49(1):44–50, 2006.
9. Jon E. Froehlich, Eric Larson, Tim Campbell, Conor Haggerty, James Fogarty, and Shwetak N. Patel. Hydrosense: infrastructure-mediated single-point sensing of whole-home water activity. In *Ubicomp '09*, pages 235–244, 2009.
10. J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. MyLifeBits: fulfilling the Memex vision. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 235–238, 2002.

11. Simon Gerber, Michael Fry, Judy Kay, Bob Kummerfeld, Glen Pink, and Rainer Wasinger. PersonisJ: mobile, client-side user modelling. In *UMAP 2010, LNCS 6075*, pages 111–122. Springer-Verlag Berlin Heidelberg, 2010.
12. Holger Harms, Oliver Amft, Gerhard Tröster, Mirjam Appert, Roland Müller, and Andreas Meyer-Heim. Wearable therapist: sensing garments for supporting children improve posture. In *Ubicomp '09*, pages 85–88, 2009.
13. CJ Heneghan, P. Glasziou, and R. Perera. Reminder packaging for improving adherence to self-administered long-term medications. *Cochrane database of systematic reviews (Online)*, (1), 2006.
14. Dan R. Olsen Jr. Evaluating user interface systems research. In *UIST*, pages 251–258. ACM, 2007.
15. Tuula Kärkkäinen, Tuomas Vaittinen, and Kaisa Väänänen-Vainio-Mattila. I don't mind being logged, but want to remain in control: a field study of mobile activity and context logging. In *CHI '10*, pages 163–172, 2010.
16. Julie A. Kientz, Rosa I. Arriaga, and Gregory D. Abowd. Baby steps: evaluation of a system to support record-keeping for parents of young children. In *CHI '09*, pages 1713–1722, 2009.
17. Sunyoung Kim and Eric Paulos. inair: measuring and visualizing indoor air quality. In *Ubicomp '09*, pages 81–84, 2009.
18. Younghun Kim, Thomas Schmid, Zainul M. Charbiwala, and Mani B. Srivastava. Viridiscope: design and implementation of a fine grained power monitoring system for homes. In *Ubicomp '09*, pages 245–254, 2009.
19. W.K. Koh, J. Ng, O. Tan, Z. Tay, A. Wong, and M.G. Helander. Why Taking Medicine Is a Chore—An Analysis of Routine and Contextual Factors in the Home. In *Procs 1st International Conference on Human Centered Design: at HCI International 2009*, pages 452–461. Springer, 2009.
20. D. Kyriacou, H. Davis, and T. Tiropanis. Evaluating Three Scrutability and Three Privacy User Privileges for a Scrutable User Modelling Infrastructure. *User Modeling, Adaptation and Personalization*, pages 22–26, 2009.
21. M.L. Lee and A.K. Dey. Lifelogging memory appliance for people with episodic memory impairment. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 44–53, 2008.
22. Ian Li, Anind Dey, and Jodi Forlizzi. A stage-based model of personal informatics systems. In *CHI '10*, pages 557–566, 2010.
23. L. Nachman, A. Baxi, S. Bhattacharya, V. Darera, P. Deshpande, N. Kodalapura, V. Mageshkumar, S. Rath, J. Shahabdeen, and R. Acharya. Jog Falls: A Pervasive Healthcare Platform for Diabetes Management. *Pervasive Computing*, pages 94–111, 2010.
24. K. O'Hara, M.M. Tuffield, and N. Shadbolt. Lifelogging: Privacy and empowerment with memories for life. *Identity in the Information Society*, 1(1):155–172, 2008.
25. Abigail J. Sellen and Steve Whittaker. Beyond total capture: a constructive critique of lifelogging. *Commun. ACM*, 53(5):70–77, 2010.
26. Tammy Toscos. Using data to promote healthy behavior in children. In *IDC '10: Proc 9th Intern'l Conf on Interaction Design and Children*, pages 344–347. ACM, 2010.
27. K Upton and J Kay. Narcissus: interactive activity mirror for small groups. In *UMAP09, User Modeling, Adaptation and Personalisation*, pages 54–65, 2009.

ISBN 978-1-74210-209-2

School of Information Technologies
Faculty of Engineering & Information Technologies
Level 2, SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.information@sydney.edu.au
sydney.edu.au/it