Relating Personally Relevant Social Information from Heterogeneous Sources

João Guerreiro, Tiago Guerreiro, Daniel Gonçalves
INESC-ID / IST / Technical University of Lisbon
Av. Professor Cavaco Silva, IST Tagus Park, 2780-990 Porto Salvo, Portugal
jpvguerreiro@gmail.com, tjvg@vimmi.inesc-id.pt, daniel.goncalves@inesc-id.pt

ABSTRACT
Proliferation of online social applications and platforms has generated enormous amounts of information that could be helpful to the users. However, this information is sparse and hard to integrate. We present a framework that inter-relates information from different online sources and, with the help of a user's personal information, is able to provide useful and relevant information from his perspective, in an iterative information seeking process. Information retrieved from the users' devices, due to its personal and trustable character, works as a filter to information retrieved from other less trustable and structured sources. We defined a single structure to inter-relate the information as a coherent whole, instead of separate chunks. To evaluate our approach, we present an application that obtains relevant information about people. The results, analyzed together with the users, suggested that it is possible to obtain relevant and inter-related information about someone, resorting both to personal and public information.

Author Keywords
Information Inter-Relation, Online Social Information, Personal Information, Iterative Search Process.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous; I.2.4 Knowledge Representation Formalisms and Methods: Semantic networks; H.3.3 Information Search and Retrieval: Information filtering, Search process; H.5.2 User Interfaces: Evaluation / Methodology

INTRODUCTION
The world-wide-web is a gigantic "information universe", one where probably lies the answer to any of our doubts or curiosities. Search engines can point the way to information about almost everything (persons, events, institutions, etc.). Sources like Wikipedia and similar sites provide additional in-depth information about a variety of subjects. Also, in the blogosphere people describe their entire life, their work, personal matters, their children, among several other subjects. In online social networks users expose their interests, preferences and work-related information even with strangers. Many people keep only a close relationship with a small group of their social network "friends", adding others for being friends of a friend (FOAF), having similar interests or just to increase their connections count. All of these online sources reinforce the idea that the Internet contains many unknown or hard to recall information, which can help users in several situations of their daily life.

However, this data is spread out by multiple applications and platforms, and if the users need to obtain information about someone or something, they have to search in each one of these sources. The time the users lose with this searching process is precious, and ways to automatically collect the desired information are sorely missed. It is inevitable to have to filter the relevant information from a multitude of general-purpose or general-interest results in order to find what the user is really interested in, given a particular context, interests and taste.

Processes for automatically extracting personally relevant information from public sources face the challenges of dealing with an enormous amount of data and the ambiguities therein. For instance, if a user wants to search for information about someone there can be many people with the same name. The user, however, wants results about a specific person and might not be interested in everything that can be found about that person. Rather, only some particular aspects, stemming from similar interests, for instance, might be relevant. Using current solutions, search about some person would yield the same results for every single user. Since important subjects for some user may not be relevant for others, it would be very useful that the search results could be based on the users, their profiles, and on what is actually important for them.

A good way to get personally relevant results is to use Personal Information as a dinstinguishing factor. Our devices (personal computers and mobile devices) are aware of most of our interactions, and have access to an enormous amount of data about us, our activities and our acquaintances. Documents, e-mails, calendar, SMSs, phone calls and even face-to-face interactions are great examples of this unique knowledge. All the data found in these (and similar) sources constitute a user's Personal Information.

The personal information in the users' devices can be used to filter and guide the retrieval of public information. Together, they will help the users to get information about people, events, subjects and places that are personally relevant to them instead of generic, one-size-fits-all, search results. Different users have different interests and different types of relations with others, so it is very unlikely that the same results please all of them. What is relevant for some user may not be for others. It is essential that the results are filtered based on the user's interests and what is relevant to them. The credibility and individual character of personal information can help filter the information from online sources, resolving the ambiguity inherent to them and presenting results with meaning and interest to the user. Knowing that all the mail messages the user exchanged with a friend were about soccer and mobile devices, it is possible to disambiguate results and direct the searches to these subjects, because those are the most important ones between the user and his/her friend. For instance, if other user with the same friend hates soccer and exchanged mail messages about music, it is very unlikely that he finds the information about soccer relevant. But on the other side, he would like to get some information related to music.

However, having multiple information sources with different structures, and some not structured at all, it is essential to find a single representation in order to easily inter-relate the information as a coherent whole, instead of separated chunks.

We present an approach and underlying framework that goes beyond the state of the art by inter-relating the users' personal information and interactions with data from online public sources. We use an iterative process of data discovery that harmonizes data from different online sources, using personal information to filter and better understand it in context. All personally relevant information is stored as an interconnected and consistent whole in an RDF-based semantic network. We can establish relations among different types of information and even reinforce the confidence of replicated information from different sources.

While laying the ground work for applications in several contexts and domains, we focus on searches about persons, resorting to personal information existent on users' devices and on data from social networks, blogs and search engines. Our evaluation shows that it is possible to retrieve, inter-relate and present relevant information about someone, from the users' perspective, resorting to personal and public social sources.

In the next section we discuss some applications that, somehow, try to use public social information and/or personal information to improve the users' experience. Then, we present our approach and the concrete framework, focusing on the inter-relation of multiple sources. Finally, we present an example web application that retrieves information about a person, as well as the results achieved in the user evaluation.

RELATED WORK

There are applications that, resorting to social web sources or personal information, try to help the user getting information about someone. A great majority of the research in this area focuses on obtaining the users' contacts context, to be aware of the right time to make a call, or to select the best communication channel. Context information often considered is related to location, contacts' social interactions [8], instant messaging application status, device state [2], idle time, schedule information, among others. These applications are only usable as long as the users allow and want their friends to be aware of their context information. Also, it is limited to a restricted group of people. Context Application [5] considers a contact as an information repository, which should be used to add context or personal information (photos, communication history, etc). It uses that information, for instance, to create a Top Contacts group based on the communication history. It has the great advantage of enriching the users' contacts repository with information from external online databases, such as the Yellow Pages. As a great example of applications which make use of personal information from the users' devices, "Forget-Me-Not" [6] records social interactions, together with sending/receiving documents, and can recall that information upon user request, working as a memory aid.

Another approach to obtain information about someone is explored in "WhozThat?" [1], by resorting to online social network profiles like Facebook or MySpace, for interest-matching. On the other side, there are some approaches that try to reconstruct a social network by analyzing the co-occurrence of names on Web pages using search engines. ReferralWeb [4] is one of the pioneers in this category and is able to estimate the relation strength between two persons or finding a path between them. Flink [7] adds information from mail messages and self-created profiles, which can help disambiguating the Web-mining count. However, these applications are not designed to get information about a person, unless related with the type of relationship.

What all the above approaches lack is making use of the enormous quantity of information sources to help the users with summarized and inter-related in-context information. Furthermore, little attempt is made in trying to combine personal information with that from public sources. This knowledge could be used to help the user anytime, anywhere, relating, summarizing and providing important data when it is required, in a personally relevant way. In a social environment it is natural to wonder "I know that person, but where from?" or "Tomorrow it is Peter's birthday, I should give him a present. What are his bigger interests?". Our system is able to provide answers to those questions by gathering and interrelating information from the user's devices enriched with public information sources, to offer the user context- and personally-sensitive information when it is needed.

SCENARIOS

To provide a better understanding of our challenges and goals, we outline a set of possible scenarios as well as the main concepts underlying our approach. The framework developed in this research context aims to help the user in several different situations. We describe some meaningful ones:

I am at a party and I find someone that seems familiar talking to a friend of mine. Using his Bluetooth ID, I ask the system about our past interactions. I get the information that we were together two years ago at the IUI Conference, and had exchanged 2 mail messages and a document (that I can access if I want to). The document's subject is also shown, as are the people I have forwarded it to. I ask for more information about him. I keep drinking my gin and see that he has interests in accessibility and loves cars; his wife is Maria Lee and was in a conference in Japan a week ago.

I am at a meeting and my boss says that, after the last break, we will discuss the case about John Terrence. Apparently he does not know that I have no idea who he is, but I see one opportunity to look good anyway. I search my new mobile phone application for him and the results show me that my friend Harris sent me and John some mails about his PhD that involves blind people mobile device usage. With that, it also shows the name of a foundation he works with and some information about the recent work he has done. After seeing the results I have an idea about who he is, but to a better preparation I call my friend Harris which apparently knows him better.

MANAGING INFORMATION FROM MULTIPLE SOURCES

There are many information sources that we can employ to help the user seeking relevant information. The world-wide-web provides us multiple applications and platforms that can be very helpful when we are trying to get information about a particular person.

Personal Content-Filtering of Public Sources

We are dealing with two different types of information: personal information available in our devices; and public information from online sources. Although the users' devices provide reliable and relevant information from their point of view, with a proper meaning to them, public sources generally are much more ambiguous. As an example, if a user searches for information about *Tony Parker* on his devices, it is most likely that he/she finds information about a single person (in a few cases there may be two, but hardly more). However, if the user searches for the same person in a search engine there will be thousands of results relative to several different *Tony Parkers*. How can the user know who is the "right" one? Also, there can be a lot of information about the same person, how can the user get information about the subjects that really matter to him/her? The personal information retrieved from the user's devices, is the perfect candidate to help filtering the ambiguous information that public sources provide. Thus, it is possible to identify which data is related to our search and at the same time collect relevant results from the user's point of view. On the previous example, it would be possible to filter the data about that specific *Tony Parker*, on subjects that really matter to the user. If the user shared some mail messages with him about politics and mobile devices, that is most likely to be the desired information rather than about rugby or football.

Some existent applications rely on other users to provide their personal or contextual information, when trying to know something about them. This can be a problem in two specific ways: First, it is very difficult for someone to accept that others have access to their information; Second, this is only reasonable in an intimate social context (if we share our information, it has to be with a very close friend). To avoid privacy issues and dependency on what others may provide, we consider only the personal information existent in the users' devices (personal computers and mobile devices) and public sources of information.

Inter-related and Coherent Information

Our approach makes place for several possible information sources. Most past and recent applications deal only with one type of information, and those who deal with more do not establish any relation between them. Each information source has a single representation, and due to this heterogeneity it is essential to find a single structure capable of dealing with the different kinds of sources, inter-relating the information and representing it as a coherent whole, instead of isolated chunks. This would turn the process of finding information about something and its references much easier and general. For instance, if we need some information about some person we could search our mailbox trying to find some e-mail that talk about him/her; search on Google and Wikipedia and finally search in my SMS inbox. In possession of a unified integrated index, we could search only for that person, immediately obtaining information from all the above sources inter-related as a coherent whole.

Like aforementioned, there are many sources that we can use in the user's benefit and it is impossible to know when a new one will appear. Considering this, it was crucial that introducing a new information source could be easily done, without changing and compromising other modules.

A lot of information can be retrieved from these sources, so there was the need to keep and structure all that information equally (either from users' devices or online sources). A special attention had to be given to non-structured sources, so the information could be converted to the common representation. Since there is data more relevant, precise or credible, there was the need to rate it, so the user (and the system itself during his iterative searching process) could have that perception.

Due to these differences in data precision and credibility, and as previously mentioned, personal information in the users' devices, being more credible and relevant from the user point of view, can be used to filter information from more ambiguous sources (mainly online). To accomplish this we had to elaborate a process of continuous iterations,

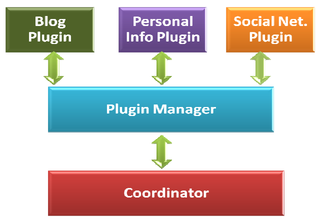


Figure 1 – High-Level Architecture

so the information could be reevaluated and reweighted. Thus, the information that really matters to the user can be presented to him. To accomplish these challenges we built a framework with an architecture divided in three main modules, detailed in the following section.

Architecture

The framework architecture is based on three main components: Plugins; Plugin Manager and Coordinator (Figure 1). Plugins are responsible for extracting the data from the different information sources and structure it into the common representation. The non-structured information is marked by the plugins, so it can be identified and sent to the Natural Language Module by the Coordinator. These plugins register in the Plugin Manager, which is responsible for selecting which plugins are suitable for each search. The Coordinator is responsible for requesting information from the plugins, store the results in a knowledge base, and iteratively requesting more information from the different sources to clarify or reinforce some knowledge.

Plugins

One of the most important assumptions of our approach relies in the ability to access personal and public information. To feed the system with this essential data, the system features a plugin based architecture corresponding to the different information sources. Plugins are the direct contact with those sources and each one of these plugins inherits from a single entity due to the similarities and shared properties between all of them.

Different sources have different structures and representations, so the information retrieved from each plugin needs to be transformed to a single one, to simplify further information processing. To accomplish this, each plugin has an adapter where it sends the information to be processed and transformed into the unique representation. With this, each plugin is able to produce structured information or tag it as unstructured (subject to further contextual processing by the natural language module).

The single representation used is a list containing tuples with characteristics of the information found. These features are: subject (search); predicate (relation); object (information); a weight assigned to that tuple; and the information source. The weight is the confidence the plugin has on that piece of information ranging between 0 and 1. These values depend on the credibility and relevance that piece of information could have to the user. For example, if the users find information about some person's interests and favourite TV shows, it is understandable that the interests are more relevant to them. Also, if an information chunk needs further natural language processing, it has less credibility, so less confidence. The weight is not relevant when we desire to store concrete properties of a given object, because they are not taken as probable or improbable. For instance, the tuple referring that *Tony Parker* sent me "*DocumentX.pdf*" needs to be weighted; however, the size, extension and path of the document do not need, because they are properties of that document.

The biggest advantage on a plugin based system is that it eases the addition of new plugins, therefore easily extending the system with new information sources. If a new source is found, it is only required to add the self-contained code of that plugin, without changing the other parts of the architecture. As the iterative search process and the Plugin Manager base their decisions on the capabilities declared by the plugins upon registration, integration with the remaining information sources is always assured. In that matter, it is only necessary to choose the plugin main capabilities upon registration.

The Plugin Manager's main task is to decide which plugins should be called, considering the current search. To accomplish this, it has to know which plugins can provide useful and relevant information, and avoid making repetitive and useless searches. Each plugin has to register in the Plugin Manager informing what kind of information they can obtain.

Coordinator

The main module of the system is the Coordinator which is responsible for requesting information from the plugins, store the results in the knowledge base, and iteratively requesting more information from the different sources to clarify or reinforce knowledge. Ultimately, it will gather the data with higher confidence levels and send it to the user. Indeed, without this module the information in the knowledge base would grow uncontrollably in every iteration and would retrieve everything in every search.

One of this module's main tasks is to ensure the storage of the information received from the plugins in the knowledge base. While some information sources are able to produce structured knowledge, others are likely to need some extra consideration. To this end, the framework features a simple Natural Language module, one that is able to look into a text and try to retrieve structured information from it. The coordinator is responsible for deciding when this needs to be performed, based on the type of reply given by the plugins (structured or not structured).

Structured information is based on tuples (Subject, Predicate, Object, weight, source) due to our requirement of small and concise/synthetic pieces of information, instead of big paragraphs that would take longer to understand with much useless information. The weight allows the plugins to provide information with different confidence levels on their trustworthiness or value. For instance, considering Facebook, some persons fill their interests like an enumeration (separated by commas or newlines), while others write a text describing it. Considering the latter, as it requires further analysis, its weight will be less than in the former scenarios, because information already structured by the user is most likely to be right than information structured automatically by the Natural Language module. In this case, the application scope defines the weight but this can also be useful for a plugin to tag some piece of information as less or more trustable. Besides the weight given by the plugin to the information, the Coordinator calculates a new value, considering also the weight it gives to each plugin. It is natural that a search on a search engine has less credibility than one performed on a social network profile, due to their non-structured and structured character; and personal information (from the user's document space) is more credible and important, in the users' point of view, than that from the aforementioned sources. The calculation of the new weight consists only in a multiplication of the two values, because we want both values to count the same. We can have plugins that find their information relevant, but the Coordinator does not assert much credibility to those plugins, so the weight needs to be diminished, or vice-versa, a plugin credible to the coordinator, but less certain of its information.

The hardest task for the Coordinator is to decide what to present to the user. The knowledge base is fed with enormous amounts of information with several relations between them. However, for a particular search, the user desires a finite, concise and understandable result. What should it iterate? When should it stop? The Coordinator answers by analyzing the new information retrieved by the plugins and compare it with the existent in the knowledge base, to find out the necessity of iterating.

Knowledge Base

With an enormous amount of information that can be retrieved from several sources, there is the need to store and organize it to get real knowledge. Otherwise, the data would be spread out, and it would be impossible to extract meaningful and inter-related information. To represent the information extracted we use a Knowledge base, a semantic network which gathers all relevant information collected from the different plugins, inter-relating it into a coherent whole. Our internal representation is based on Subject-Predicate-Object relations, for example "Tony Parker-Interests in-Mobile Devices", since we need objective and synthetic information. We do not want big descriptions that take too long to read and understand, with higher detail levels. One of our framework's bases is the simplicity and atomic character of the stored and presented information. To accomplish that, the Knowledge Base uses the *Resource Description Framework (RDF)* which is a match to our needs, representing the information as triples (Subject, Predicate, Object). RDF is simple to use and allows some manipulation due to its permissions to store everything. There are no obligations or restrictions to the information represented on the Knowledge Base, so the triple subject-predicate-object can be anything we want.

The information considered is represented by resorting to two different case frames: simple or weighted. The simple case frame is used when we desire to store the properties of a given object. They are not taken as probable or improbable. They are just elements defining some entity. A simple case frame example is the characterization of a document (Figure 2), e.g., *name, path, creation date, modification date, keyword.* In this case frame, it is used the standard representation, with the subject being the node representing the document, the predicate are the different relations (characteristics) and objects are their values.

However, while the information that defines some node can use the standard representation, when we need to represent other features, it is not enough. In particular, to some information we desire to add two extra items: weight and origin. Figure 3 presents a scenario where the weighted case frame is used. In this frame, the components of a traditional relation are placed as links (predicates) between nodes and objects. This enables us to include as many features as we desire for this particular information. So, the subject would be the node representing this piece of information; the different predicates are the set subject, relation, object, weight and source; and the objects are their values.

There is the need to access the information represented in the Knowledge Base, either to present it to the user, or to verify if there is information that need to be reinforced when new information arrives. To accomplish that, we use *RDF SPARQL* query language. Similar to traditional SQL, SPARQL allows us to craft complex queries to inspect our data in efficient ways.

Iteratively seeking for information

The coordinator checks, for each main field in the previous iteration, the number of new triples with a reasonable value of credibility (for example, 0.5 from 0 to 1). If there is a considerable set of new and valuable information, it is not

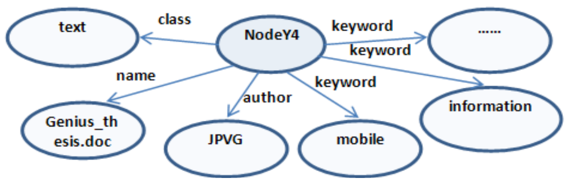


Figure 2 - Simple RDF Case Frame

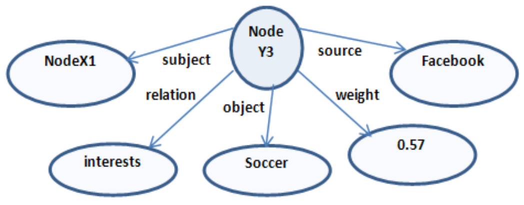


Figure 3 – Weighted RDF Case Frame

required to search again that field. On the other hand, if there are not new values (or only a few), it is worth to search again that field. All those fields are marked to being searched on the next iteration, so the Coordinator can send to the Plugin Manager the kind of information to be searched. In these iterations, the best/higher values in the knowledge base are used as additional context. Thus, highly valued information is used to improve the quality of the results and help disambiguating.

Although the previous iteration checking may select the main fields to be searched, when a lot of new information is found (in all fields), it does not mean that there is not more information useful in our sources of information (that can be improved using appropriate context). To definitely stop iterating, or to keep iterating even if our main fields are well supplied, we consider the number of new values, relating to the number of old values obtained in this iteration. The number of new values has to be bigger than ¼ (parametrizable) the old ones to keep searching. If it is smaller, there are only a few new values, so it is most probable that the next iterations will converge to zero. Hence, the ultimate decision is based on the ratio between old and new information, one that verifies if the search is converging. When this process is terminated, we need to decide which information we present to the user. Since we have all the data weighted, we can establish a limit and the one with a bigger value can be presented to the user.

Updating knowledge

Each plugin assigns a different weight to the information it extracts and the coordinator recognizes different degrees of credibility for each plugin. Also, different plugins might provide the same information about a concept, thus accumulating evidence of its truthfulness, while in other cases opposing information might result. To account for this, the semantic network allows the different relationships between concepts to be weighted, as an indicator of their credibility and at the same time associate their source(s) of information(s).

It is important to notice that the same information may be retrieved from different plugins. It is likely that the duplicated information is relevant. However, it is not trivial to reinforce the information mostly due to the diversity of forms it can be presented. We want similar values to be marked as equals, so when both appear instead of having two different values, the weight is recalculated. A good example is someone that has in his Facebook interests "Machines", and in his blog says to be interested in "Machinery". We want both to be the same, and to accomplish that we stem the information before inserting it in the Knowledge Base. Stemming gets the root of the word, so when we have similar words they count as the same. When we want to show the information to the user, we reconvert the stemmed word in the smaller word originating that stemmed one (ex: "machine"), as we are interested in the concepts and contexts and not an exact copy of the original data.

When some information chunk, already indexed in the knowledge base, presents itself again, the confidence on that piece of data is reinforced. We developed an algorithm that respects the value of the information and maintains a normalized weight scale.

Consider that a relation is previously weighted with 0.8 (in a scale from 0 to 1). A duplicated entry is detected with a weight of 0.5. The weights are recalculated as follows:

The relation is to be reinforced. To this end, the initial 0.8 are guaranteed and we are only working with the remaining weight percentage:

```
1 - OldWeight [0.8] = 0.2
```

We use the new weight (0.5 in the example) as a reference value to scale the remaining weight (0.2):

```
AddWeight = 0.2 * ArrivingWeight[0.5]  = 0.1
```

The calculated value is added to the old value:

```
NewWeight = OldWeight + AddWeight = 0.9
```

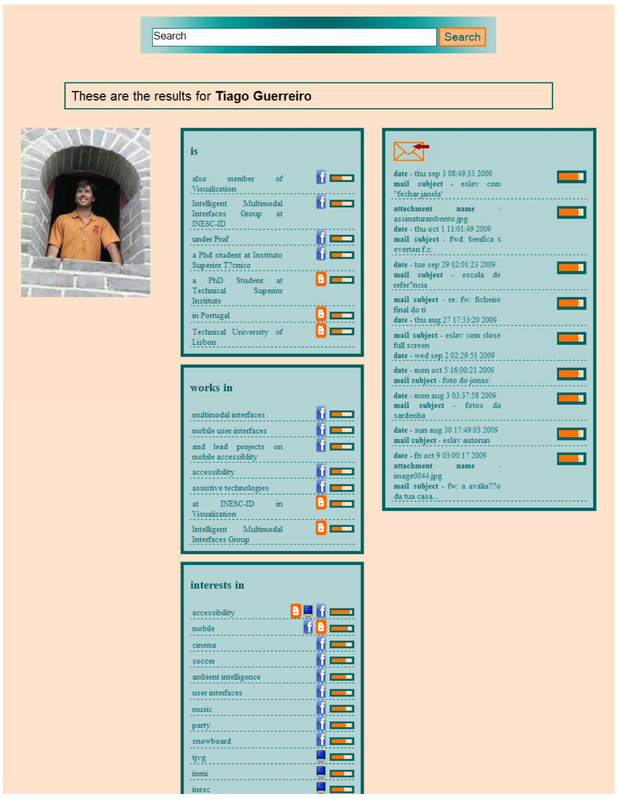
The new weight on the Knowledge Base would now be 0.9. This algorithm allows us to always reinforce the information when similar information arrives, but in a moderate and consistent percentage. Since the old weight was 0.8 and the information appears again, we always want the new weight to be bigger than the old one. So, we use the remaining 0.2 to help calculating the new value and add it to the 0.8.

When new information is added or the weight recalculated, there is a counter keeping the register of the number of new and old (duplicated) pieces of information. This helps the Coordinator in its iteration decision-making, by being aware of the old information related to the new one. If there is much more old information than new, there is no need to continue interating, because it is converging to zero.

To keep recycling and refreshing the information we use timestamps, so that older data starts losing its weight. It is important because that information, someday correct, is now incorrect or out of date. The older the information is more weight it loses. If the information is still recent (1 month or less) the weight stills the same, and being the information found on the same plugins, it does not change the current weight. However, if the information is old enough (more than one month), a new weight is calculated to decrease its value. The expression used to find the value to decrease is based on months (between 1 and 2 months count as 2; between 2 and 3 count as 3, and so on). The maximum value is 12, so information with 12 months or more has the same value (12). The expression is:

```
Value to Decrease = ln((months/10) + 1)
```

Since this expression possible month values are between 2 and 12, the values to decrease are between 0.18 and 0.79


Search

Search

These are the results for **Tiago Guerreiro**

is

also member of Visualization

Intelligent Multimodal Interfaces Group at INESC-ID

under Prof

a Phd student at Instituto Superior T?cnico

a PhD Student at Technical Superior Institute

in Portugal

Technical University of Lisbon

works in

multimodal interfaces

mobile user interfaces

and lead projects on mobile accessiblity

accessibility

assistive technologies

at INESC-ID in Visualization

Intelligent Multimodal Interfaces Group

interests in

accessibility

mobile

cinema

soccer

ambient intelligence

user interfaces

music

party

snowboard

tjvg

immi

inesc

date - thu sep 3 08:49:33 2009
mail subject - eslav com "fechar janela'

attachment name - assinaturambento.jpg
date - thu oct 1 11:01:49 2009
mail subject - fwd: benfica x everton f.c.

date - tue sep 29 02:01:23 2009
mail subject - escola de refer?ncia

mail subject - re: fw: ficheiro final do ri
date - thu aug 27 17:33:20 2009

mail subject - eslav com close full screen
date - wed sep 2 02:29:51 2009

date - mon oct 5 16:00:21 2009
mail subject - foto do jonas

date - mon aug 3 03:37:58 2009
mail subject - fotos da sardenha

date - sun aug 30 17:49:03 2009
mail subject - eslav autorun

date - fri oct 9 03:00:17 2009
attachment name - image0044.jpg
mail subject - fw: a avalia??o da tua casa...

Figure 4 – Example Application: Searching for a person

approximately. The old weight minus the decreasing value, results on our new weight (cannot be less than 0). An example, a tuple with 5 months old and a weight of 0.8, using this expression, is now near 0.4. If that information appears on the next search, which means it is still relevant, so it will increase its weight again. If it does not appear, it means that the information might be incorrect or obsolescent, so it maintains the lower weight.

EXAMPLE APPLICATION

To prove that our framework can provide relevant information to the user and satisfactory results, we developed an example application with the goal of obtaining information about a person (Figure 4). The main concern of our interface is to turn the process easy to interact with, centering it on a simple search task, but at the same time make use of all the features involving our framework. That includes the personal and public information, including its filtering, the iteration process, the data from different sources and structures (also non-structured) being represented as a coherent and inter-related whole, the decision of what to present and all it brings.

This application resorts to four different types of information sources: personal information existent on the user devices (e-mails); social networks (facebook); blogs (blogspot) and Wikipedia. We limited the sources in this proof-of-concept application to these sources as they represent different types of information, enough to prove our assumptions and at the same time keep the system evaluation simple. The users' devices (in this case, their personal computers), provide us with e-mails, which are rich interaction-wise and are likely to be useful for the user and at the same time are helpful in filtering the enormous quantity of information at public online sources (using the information therein). To access this information, we use Scribe, our personal information plugin-based monitor that indexes information from various personal sources (e-mails, documents, webpage visits, calendar, contacts) [3]. As a representative social network we selected facebook as it is widely spread worldwide and particularly within the Portuguese society. Also, it is the most used and preferred by those who tested our application. Blogspot/Blogger is also a very popular blog platform and Wikipedia has descriptions of many people, famous or important.

In this application we support searches for people with different relationships with the users. It can be a very close friend, some person they know but have not much information or even a celebrity. We chose the information sources with the intent to cover all these scenarios. Scribe (Personal Information) and Facebook are more intimate, so apply mostly to close friends, but there is also a big probability to contain information about a "known" person. On the other side, Wikipedia only contains information about important people, and Blogspot is more transversal and provides information on both these scenarios.

Application platform

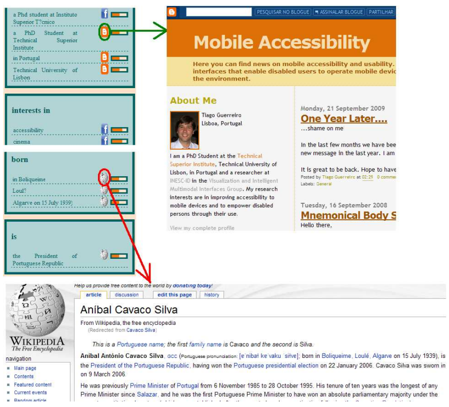
There are many contexts where our framework can be used. Using mobile devices as an entry point it is possible to use our applications almost everywhere. Besides that, it is perfectly possible that users want/need to use it in the comfort of their homes or offices. For example, if someone calls their home telephone and they are near their Laptop, it is quicker and very useful to access the application there. We found an online service accessible through a webpage as the best option to deploy our framework's applications. It is thus accessible from every device. The only limitation for this solution is the obligation to always be connected to the Internet, but this application nature makes this an obligation to all possible solutions. Also, it does not need previous installation or configurations to perform a quick search.

Interface

The interface for this application is very simple and focused on the search task. To search for somebody it is only necessary to write the person's name and click enter/search.

On Figure 4 we can see that the results are divided in three different columns. The left one is for the photo(s), to have a visual idea/confirmation; at the center the description of the person (interests, work information, birth, etc); and the right column shows the interactions they had (in this case, the mail messages exchanged).

We only show results with a confidence superior to 0.5 (from 0 to 1), because less than that is information not credible. Also, each item has a clickable icon indicating its


a Phd student at Instituto Superior T?cnico

a PhD Student at Technical Superior Institute

in Portugal

Technical University of Lisbon

interests in

accessibility

cinema

born

in Boliqueime

Loul?

Algarve on 15 July 1939)

is

the President of Portuguese Republic

PESQUISAR NO BLOGUE | ASSINALAR BLOGUE | PARTILHAR

Mobile Accessibility

Here you can find news on mobile accessibility and usability.
interfaces that enable disabled users to operate mobile devic
the environment.

About Me

Tiago Guerreiro
Lisboa, Portugal

I am a PhD Student at the Technical Superior Institute, Technical University of Lisbon, in Portugal and a researcher at INESC-ID in the Visualization and Intelligent Multimodal Interfaces Group. My research interests are in improving accessibility to mobile devices and to empower disabled persons through their use.

View my complete profile

Monday, 21 September 2009

One Year Later....

...shame on me

In the last few months we have bee
new message in the last year. I am

It is great to be back. Hope to have
Posted by Tiago Guerreiro at 02:29 0 comme
Labels: General

Tuesday, 16 September 2008

Mnemonical Body S

Hello there,

Help us provide free content to the world by donating today!

article | discussion | edit this page | history

Aníbal Cavaco Silva

From Wikipedia, the free encyclopedia
(Redirected from Cavaco Silva)

This is a Portuguese name; the first family name is Cavaco *and the second is* Silva.

Aníbal António Cavaco Silva, GCC (Portuguese pronunciation: [ɐˈnibaɫ kɐˈvaku ˈsiɫvɐ]; born in Boliqueime, Loulé, Algarve on 15 July 1939), is the President of the Portuguese Republic, having won the Portuguese presidential election on 22 January 2006. Cavaco Silva was sworn in on 9 March 2006.

He was previously Prime Minister of Portugal from 6 November 1985 to 28 October 1995. His tenure of ten years was the longest of any Prime Minister since Salazar, and he was the first Portuguese Prime Minister to have won an absolute parliamentary majority under the

WIKIPEDIA
The Free Encyclopedia

navigation
- Main page
- Contents
- Featured content
- Current events
- Random article

Figure 5 - Clickable icons and the resulting webpages

information source, and opening the concrete webpage from where it came from, providing the information context and access to more detailed information, if necessary (Figure 5).

EVALUATION

To acknowledge our approach as a success, some research questions need to be answered: 1) Can our framework inter-relate information from different sources (either personal, or public)?; 2) Can we provide useful and relevant results about persons, from the user's point of view?; 3) Can this system respond with relevant results independent of the type of relationship the users' have with the *searched ones*?

Procedure

The evaluation procedure was divided in two different phases: preparation and execution. The preparation steps included introducing the user to the evaluation, indexing personal information available in the user's computers or online services (webmail) as well as configuring other services (like Facebook) to enable users' access to their friends' profiles (using the provided APIs).

With all steps completed, all the ground work was set for the evaluation. It was composed by 6 searches performed by the users. We asked the users to choose 6 individuals with different relation magnitudes (a public figure vs a good friend or a relative). Besides wanting to evaluate the results' trustworthiness, we also wanted to evaluate how the results match the user's expectations. Thus, for each search, we asked them to write the information they were expecting to get from each one of them, the relevant information about that person from their point of view. Then, the users perform their tasks and, upon completion, answer a final questionnaire validating the results, evaluating the application by rating several system features with a 5-point Likert Scale and offering subjective feedback.

Users

To evaluate GeniusPhone, and the underlying framework and approach, we performed the evaluation with 14 users, 10 males and 4 females, with ages comprehended between 22 and 57 years (averaging 28 years old).

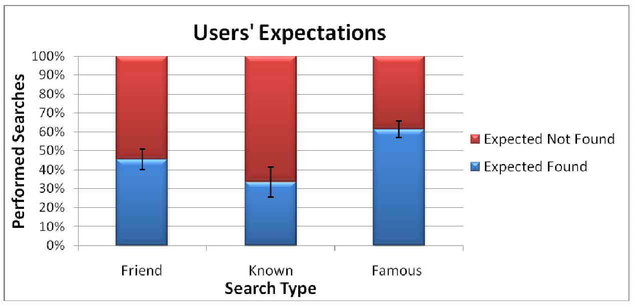


Figure 6 - Users' expectations and achieved results

Tasks

Users had to search for some people in order to obtain information about them. These searches were divided in three types (2 Close, 2 Known and 2 Famous persons), which differ in the different kinds of relationship the users have with the searched ones.

Results

To prove our assumptions and answer our research questions, after the users have performed each task, we collected the data that we found helpful to evaluate our system. Some of that data was automatically collected (total information per plugin found and shown), but other deserved a more careful analysis based on the users' opinions, approval and the information they were expecting to get from each search.

As our approach tries to provide relevant results from the users' perspective, those results had to be analyzed by them. They are the ones knowing which information is relevant, irrelevant or garbage for them. Also, we needed help from them to quantify the information they were expecting to get but was not shown. We need to know if that information was impossible to get, or our framework missed it.

Evaluating Users' Expectations and Results' Relevance. Before performing our tasks the users described what they were expecting to get from each person. This information is based on features they know about those persons and believe to define them or are somehow related to them. It is important to notice that we have not restricted this process and are not aware, before the experiment, if the information is correct or available in any of the searched sources. This data is very important so we can analyze our results accordingly to user expectations. To answer this question we have to analyze some different aspects. First, which information from users' expectations was, and was not presented? Figure 6 shows the results, in percentage, for the three different search types.

From this chart, we can observe that, regarding a Friend or Known person search type, the values are below 50%, as to Famous people the results are near 60%. Comparing Friends to Known, having more interaction with close friends it is understandable that it will improve the results, giving them a little advantage. On the other side, Famous searches have more available information on the Internet,

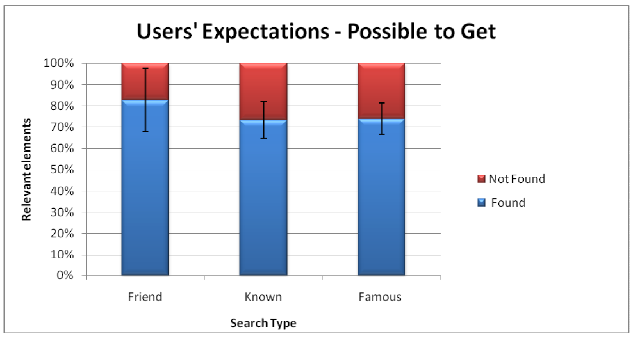


Figure 7 – Results for the information possible to get

and on the searches we used, there was always information to be found on our information sources.

Although these results are not outstanding, on most scenarios, mainly respecting to Known and Famous people, some relevant information is enough to enlighten and help the users through their difficulties. Besides, it is important to remember that the users' expectations were not pre-processed in any way. Also, we can observe on Figure 7 that if we consider only the information that was possible to find using our sources, the results improve substantially. This information was verified with the users in a post-test analysis. This indicates that, from the information users were expecting, only a small part was accessible to us but somehow we missed it.

Figure 8 shows the relation of relevant information elements found that were and were not expected by the users (average values). It was possible to verify that it presents more relevant information not expected than the number of the users' total expectations. Although users were not expecting this data, they tagged it as relevant in a post-test analysis and found it to be useful. Also, our approach fits on scenarios that the information users need is the one they do not remember at all, so they could not be expecting it. These results suggest that, respecting to Friend and Known searches, although not presenting all the expected information, the expectations are exceeded and show relevant information that the user did not remember.

Analyzing information sources. Figure 9 shows that blogs have a minor contribution to the total results, though it has a good success rate. Friend and Known searches are dominated by information from Scribe and Facebook, with a bigger predominance to the first (near 55%). On Famous searches, the relevant results belong totally to Wikipedia. This is perfectly understandable since users did not have any interactions with them.

Search Types. Searches performed for a Famous person present better results when compared to the users' expectations, but on the other hand, present less relevant information that was not expected. Indeed, the expected information and retrieved one are very similar in this scenario. Friend and Known searches' results suggest many similarities between these two groups. However, it is

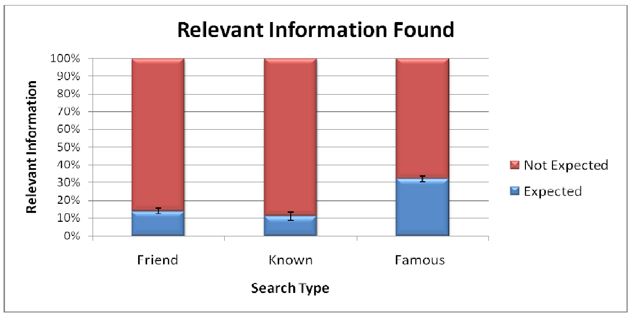


Figure 8 – Relevant information presented to the user

relevant to mention that Friend searches normally present better results. This search type behaved better relating to expected information and presents more relevant information outside the expected range. Also, considering the information not found that was expected by the users, most of the items (greater percentage in Friends than in Known) were not possible to retrieve as they were not available (as verified with the users by post-checking the sources). This combination of results suggests that when more interaction happens, and consequently more personal information is shared, the results are better.

Filtering with Personal Information. As some users had friends or "known persons" in common, we used that to compare their searches and check if their personal information influenced the public information search results, by directing the searches. We could observe that those who retrieved more personal information could guide their searches and obtain relevant information from public sources, i.e. blogs. These results could only be obtained using two (2) iterations, as in the first one, for all those searches, nothing was found on blogs. Those which could get relevant personal information used it as context to help filtering on other sources. The result was additional relevant information, which could not be obtained by the users that had no interactions or connections with them.

User's Opinions. We are pleased to notice that users found our system easy to interact with and attractive (4.7 and 4.1 averages on a 5-point Likert scale, respectively). More important are the ratings to the results' usefulness, understandability and relevance, which were rated with average values between 4 and 4.1. These ratings suggest

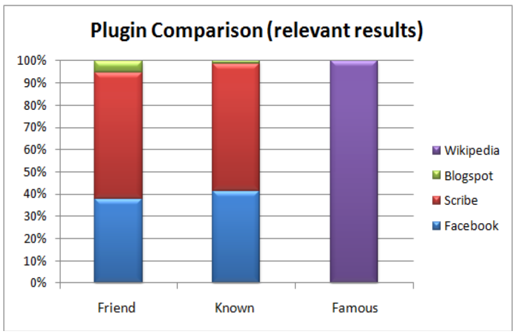


Figure 9 - Plugins influence on the number of relevant results

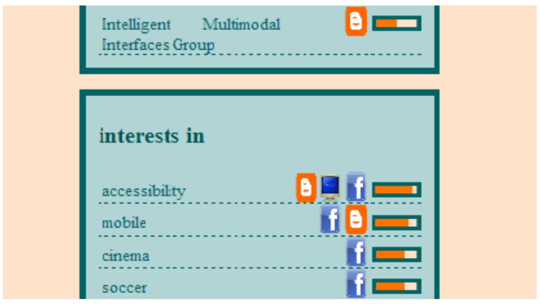


Figure 10 - Different sources reinforcing the same information

that users were satisfied and found our application helpful.

Discussion

In this section, we analyze the results taking into account the aforementioned research questions.

Except for Famous searches, the results are a combination of information from different sources. Personal Information and Facebook are more predominant but that is related to the fact that almost everyone had Facebook and exchanged e-mails with the users. However, the results represent a unified and inter-related whole of information, instead of separated chunks, gathered from different types of online social sources and users' devices.

During task execution it was possible to realize that, besides presenting the information as a whole, it was also managed as a whole. This could be observed since the same information, even from different sources, was treated and shown as the same, in a unified way. Also, this inter-connection allows the information to be reinforced and gain preponderance as a result and as context for further iterations. Figure 10 presents an example where a piece of information is shared by many sources, and although only the interest in *accessibility* is marked with Scribe's icon, the remaining information could not be found or reinforced without it. It was the Personal Information found on a first iteration that allowed us to find the information from blogs on a second one.

Results also suggest that, in average, the searches provided relevant information. Although some expected information was not shown to the users, it was majorly impossible to obtain. Also, the suggestion that it presents more information that was not expected adds great value to our results, since for most scenarios the information users do not remember is the most useful for them.

In every chart we separate the searches based on the different scenarios to analyze if our framework is suitable for all of them. We can observe that besides the different results on the different charts, the three search types present good results.

CONCLUSION

Trying to get more information about someone is a recurrent task for many users. Nowadays, there are multiple online resources where people expose their life, interests and work data, which could allow other people to discover or recall relevant and useful information. However, the available amount of information is enormous and to be useful it must be contextualized and summarized.

Our approach gathers personal information from the user's devices, and use it as a filter to the information available in public sources like search engines and social networks. After an iterative process of searching, renewing and improving the information retrieved, from the user point of view, it is able to present contextualized structured information. An example application, evaluated with users, was presented as valuable reaching the aforementioned personal and online social information access, inter-relation and coherency goals.

The plugin based architecture allows us to easily extend the framework, so in the future we plan to explore new information sources. Particularly, we will extend our web search plugins, adding more social applications and the ability to recognize relevant chunks of information about persons, particularly in personal web pages. Using this platform, there are several scenarios and applications that can be explored, so we will focus also on trying to find new uses to it, mainly in mobile contexts.

ACKNOWLEDGMENTS

João and Tiago Guerreiro were supported by the Portuguese Foundation for Science and Technology, grants SFRH/BD/66550/2009 and SFRH/BD/28110/2006.

REFERENCES

1. Beach, A., et al.. WhozThat? Evolving an ecosystem for context-aware mobile social networks. Network, IEEE, 22(4), pp. 50-55, (2008)
2. Begole, J. B. et al. Lilsys: Sensing unavailability. In Proc. ACM conference on CSCW, pp. 511-514, (2004)
3. Gonçalves, D. (2007). Narrative Interfaces for Personal Document Retrieval. PhD thesis, Instituto Superior Técnico.
4. Kautz, H., Selman, B., and Shah, M. Referral Web: combining social networks and collaborative filtering. *Commun. ACM* 40, 3 (1997), 63-65.
5. Jung, Y., Anttila, A., and Blom, J. (2008). Designing for the evolution of mobile contacts application. In Proc. MobileHCI , ACM (2008), 449–452
6. Lamming, M. et al. "Forget-me-not: Intimate Computing in Support of Human Memory. Proc. of Next Generation Human Interfaces, (2004)
7. Mika, P.. Flink, Semantic Web technology for the extraction and analysis of social networks, J. Web Semantics 3 (2005) 211–223.
8. Raento,M. et al. "ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications," IEEE Pervasive Computing(2005), vol. 4, no. 2.