intelligent context-sensitive interactions on desktop and the web

Alan Dix[1], Tiziana Catarci[2], Benjamin Habegger[2], Yannis Ioannidis[3], Azrina Kamaruddin[1], Akrivi Katifori[3], Giorgos Lepouras[3,4], Antonella Poggi[2], Devina Ramduny-Ellis[1]

1. Computing Department
Lancaster University, Lancaster, UK

2. Dipartimento di Informatica e Sistemistica
Universita' di Roma "La Sapienza", Rome, Italy

3. Department of Informatics & Telecommunications
University of Athens, Athens, Hellas (Greece)

4. Dept. of Computer Science and Technology,
University of Peloponnese, Tripolis, Hellas (Greece)

alan@hcibook.com, catarci@dis.uniroma1.it, benjamin.habegger@dis.uniroma1.it, yannis@di.uoa.gr, a.kamaruddin@lancaster.ac.uk, vivi@mm.di.uoa.gr, gl@uop.gr, antonella.poggi@dis.uniroma1.it, devina@comp.lancs.ac.uk

http://www.hcibook.com/alan/papers/avi2006-context/

ABSTRACT

In this paper we describe briefly three systems: onCue a desktop internet-access toolbar, Snip!t a web-based bookmarking application and ontoPIM an ontology-based personal task-management system. These embody context issues to differing degrees, and we use them to exemplify more general issues concerning the use of contextual information in 'intelligent' interfaces. We look at issues relating to interaction and 'appropriate intelligence', at different types of context that arise and at architectural lessons we have learnt. We also highlight outstanding problems, in particular the need to computationally describe and communicate context where reasoning and inference is distributed.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation (e.g., HCI)]: User Interfaces – Theory and methods, User-centered design.

H.1.2 [Models and Principles]: User/Machine Systems – Human factors.

General Terms

Algorithms, Design.

Keywords

Context, human computer interaction, natural interaction, user experience, dynamic interaction, intelligent interfaces

1. Introduction

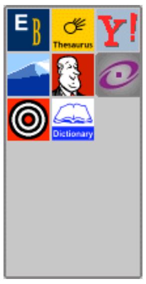
In this paper we describe briefly three systems that embody context issues to differing degrees, and use these to exemplify some more general issues about the use of context and how to use contextual information in 'intelligent' interfaces.

The systems we will look at differ in their level of development: the first, onCue, was developed and distributed commercially a few years ago; the second, Snip!t, is currently available as a web-based tool, but is still under development; and the third, ontoPIM, a is still at design stage.

2. Three Systems

2.1 onCue

onCue was produced and distributed by aQtive, a dot.com company that traded between 1998 and 2000 [Dix, 2000]. onCue is a form of intelligent toolbar, sitting at the side of the screen and watching for changes to the clipboard (through copy–paste). When the clipboard changes, onCue would alter its icons to suggest additional things that the user might like to do with the clipboard contents. For example, if the user selected a person's name various web-based directories would be suggested, if instead a table of numbers were selected, graphs and spreadsheet options would be suggested. The icons did not suddenly change, but instead slowly faded in and out over a period of about 1 second in order to not distract the user.



onCue was context sensitive in the sense that the toolbar altered depending on the users current focus insofar as this was exposed by the most recent copy/cut–paste. However, it did not demand attention in that the slow fade of icons would not distract the user and, more important, unlike the Microsoft paper clip, it was not modal, getting in the way of typing.

The internal architecture of onCue consisted of two main kinds of components: recognisers and services (see Fig. 1) linked by a blackboard-like infrastructure. The recognisers examined the clipboard contents to see if they were a recognised type (postcode, name, table, etc.). The services instead responded to data of particular types (e.g. single word for dictionary, post code for mapping web site) and were activated when clipboard contents were recognised to be that type.

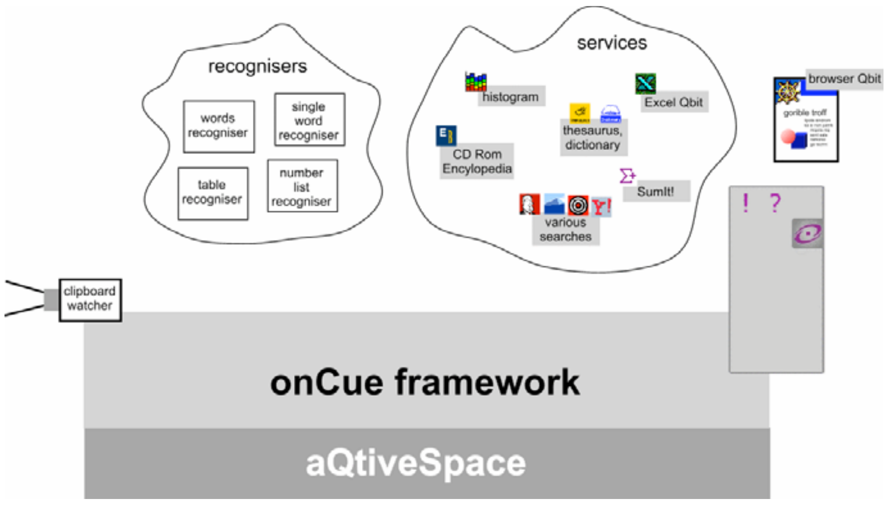


Fig 1. **onCue architecture**

Around the same time as onCue's development, there were a number of other data detector projects at Intel [Pandit, 1997], Apple [Nardi, 1998] and Georgia Tech [Wood, 1997]. More recently data detectors hit the headlines in disputes about Microsoft SmartTags. However, there is another older collection of work that started nearly 10 years earlier, again in the HyperText community, where notions of external linkage were important and Microcosm [Hall, 1997] developed at Southampton pioneered automatic links. This used an index of key terms attached to a particular content. When the user viewed a document any key terms in the index became live links. Note that most of the data detector's work, including onCue relied on largely syntactic/lexical matching using regular expressions or other patterns whereas Microcosm was lexicon based.

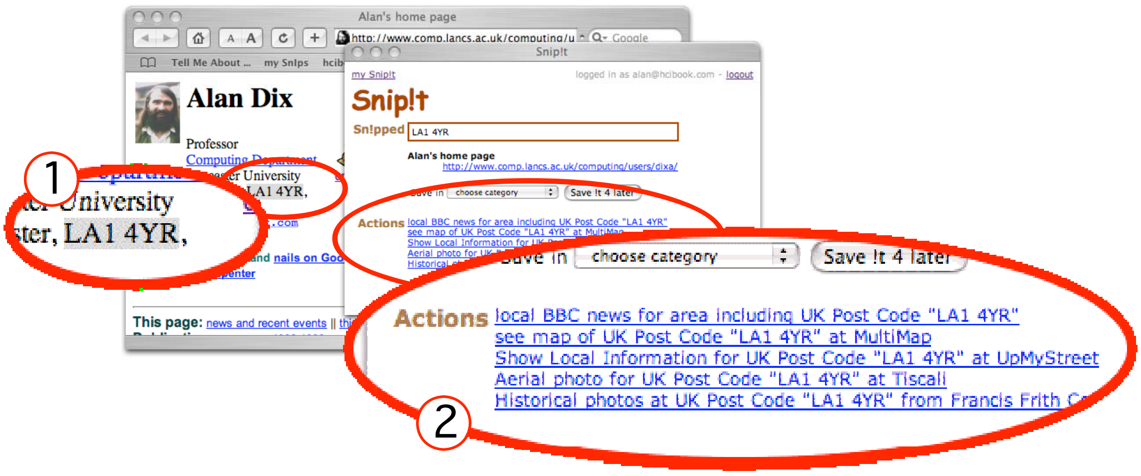
2.2 Snip!t

Snip!t is a web bookmarking tool, however unlike most web bookmarks, the user can also select a portion (snip) of the page content and then, using a bookmarklet, this selection is sent to the Snip!t web application. The snip of the page is stored along with the page url, title etc. and the user can sort the snip into categories or share it with others using RSS feeds.

In addition, Snip!t has onCue-style features – when the selected text is a recognised type of data such as a date, post code, person's name, etc., then actions are suggested. For example, in Figure 2, the selected text "LA1 4YR" (1) is recognised as a post code and this leads to suggested actions such as finding the local BBC news for the area (2).

Snip!t uses a similar architecture of recognisers and services to onCue, but these are server based rather than client based. Consequently, Snip!t can use large lexicons hosted on the server (e.g. comprehensive world gazetteer) alongside syntactic rule matching. For example, the bible verse recogniser uses a small lexicon of bible chapter names to trigger a syntactic rule to match chapter and verse numbers.

Given in this case the post code is the selected text, Snip!it is being 'intelligent', but not context sensitive. However, the intention is to make Snip!t more context sensitive in the kinds of types it recognises and the actions suggested. A good example of this is the name detector. If one selects "Alan Dix", then this is correctly recognised as a name, and internet directory services are suggested. In addition a link is given to lookup "Alan Dix" in IMDb … Snip!t does not know that Alan Dix is not a movie star! For related reasons, a telephone number detector has not yet been implemented – whilst it is easy to recognise the right number of digits, it is unclear which country the telephone number is in. The intention is to use additional technology developed at aQtive that can classify the



LA1 4YR

Alan's home page
http://www.comp.lancs.ac.uk/computing/users/dixa/

Actions local BBC news for area including UK Post Code "LA1 4YR"
see map of UK Post Code "LA1 4YR" at MultiMap
Show Local Information for UK Post Code "LA1 4YR" at UpMyStreet
Aerial photo for UK Post Code "LA1 4YR" at Tiscali
Historical photos at UK Post Code "LA1 4YR" from Francis Frith Co

Fig 2. **Snip!t in action – actions for a post code**

kinds of material on a web page. So, if a name is on a page with lots of words relating to films, then IMDb might be suggested; but if it is on a page that looks like academic computer science then DBLP lookup is more appropriate. Similarly if a telephone number is on a page from an ".it" domain name it is likely to be an Italian telephone number whereas if it is ".uk" it is likely to be British.

This also works at the level of recognisers. Currently the bible verse recogniser would think that "I gave John 3 apples" included a reference to the bible chapter "John 3". Possibly this would be distinguishable by natural language rules; however we can perform simple context-related checks. If the page includes references to other less ambiguous bible verses (e.g., Habakkuk) or the user frequently clicks on bible verse links then the bible verse recogniser is more likely to be invoked for the ambiguous "John 3". On the other hand, if the page does to appear to be related to the bible and the user never looks at this sort of material, then "John 3" is more likely to be just a person's name that happens to be followed by a digit.

Currently Snip!t only deals with pre-programmed recognisers and services. The intention is to enhance it with a means of adding user-defined and user-shareable components, as was the case with onCue. This has some interesting technical issues in that the mechanism needs to be easy for naïve users, but powerful enough for experts … and needs to protect both end users and the central Snip!t server from malicious code!

The CREO system [Faaberg, 2006], a recent web-based data detector leveraging semantic-web technology, allows the user to train actions by example. The Snip!t system does not support this type of interaction currently, and because of its largely server-based architecture, the capability for this would be limited without an associated browser plug-in.

2.3 OntoPIM

As part of the EU DELOS Network of Excellence, the TIM project (Task-centered Information Management) is studying the potential for users to store files, email, etc., indexed by personal ontologies. Design and implementation of the prototype tool OntoPIM [Katifori, 2005] is still in early stages but several key issues are already apparent.

One problem is with initial storage and classification. It is hard to save things in a simple hierarchical file system, so at first it seems unlikely, however useful it might be later, for a user to take the trouble to effectively store items in an organised fashion. However, we believe that relatively simple matching and automatic classification may help. For example, if a document contains an email address then the system can check if this is a known email and if so suggest associating the document with the person. Alternatively, if the document is an email from a person X and contains a telephone number, but no other name, then it might suggest associating the email with the relation "telephone number of person X".

One of the aims of ontology--based storage is to make task-based interactions easier. For example, if Antonella has recently received an email from Yannis, then the person "Yannis Ioannidis" will have a high activation level in Antonella's personal ontology. If subsequently Antonella invokes a route finding service then the address of the person "Yannis Ioannidis" (if known in the ontology) would be the default in the relevant address entry field.

Suppose now that the email that Antonella received from Yannis contained information about travelling to Athens, which Antonella needs in order to participate in the next meeting of the TIM project. By saving the email received from Yannis in the OntoPIM system, the information in the email is also stored in the Personal Ontology as the travel dates and the travel destination. If she later wants to book a hotel for the meeting in Athens, she needs to access the Personal Ontology to obtain this information. Let us assume that Antonella has different accommodation requirements, depending on whether she is travelling for business or pleasure. The OntoPIM interaction should be context-sensitive in the sense that the system should recognize that the information Antonella is accessing comes from an email concerning "business". This information is actually contained in the system since the email was sent from Yannis, who is a member of the TIM project, and copied to other TIM members. Therefore, when providing Antonella with the capability of booking a hotel in Athens via a Web Service, OntoPIM should recommend only those hotels that fit the appropriate requirements.

3. General Issues

3.1 Appropriate Intelligence

There is a great difference between the onCue “context" where the system autonomously inspects the clipboard to suggest actions and Snip!t where the user explicitly selects text. The former is a kind of *incidental interaction* [Dix, 2002] where the users intention is focused on one thing (copying text) and 'incidentally' some other system action occurs (onCue changes its icons).

In all types of intelligent interfaces, but particularly those where the system takes initiative within the interaction, it is important that the system embodies principles of *appropriate intelligence* [Dix, 2000] – that is embedding the intelligence within an appropriate interaction framework. A key aspect of this is to supplement the usual goal of intelligent interfaces "try getting things right as often as possible and doing something good". While it is good in demos to show that the system *can* give some added-value it is not the thing that is most critical in practice.

In fact systems can be fairly simple and indeed error prone and furthermore deliver only marginal benefit, so long as they follow the key tenet of appropriate intelligence … "when things go wrong … don't mess up the user". For example the Microsoft Office paper clip can say quite useful things when it is right, but if it is wrong it has interrupted your typing and spoilt your chain of thought. In contrast the Excel sum (Σ) button uses simple rules to choose a default range for the sum, but if the chosen range is wrong one just selects the correct one with little more effort than if the default had not been there. In addition the user has explicitly invoked the sum, so is at a natural point in the interaction to verify the selection and not be interrupted.

onCue was designed with appropriate intelligence in mind. This is why (a) it is not modal, so it does not interrupt one’s current actions and (b) its icons fade in slowly, so it does not distract one’s attention. When onCue's answers are not useful it does not force itself upon you.

3.2 Types of context

Although somewhat overused, we can look at context using who–what–where-style categories:

WHAT	data/text in the user’s current focus. e.g. clipboard, selected text.
WHERE	immediate environment – for example, if the document to be saved in ontoPIM is an email, we can use the sender, perhaps related emails with similar subject lines as context. In the case of Snip!it, as we have discussed, we can use the

country of origin or the topic of the web page to disambiguate telephone numbers or names.

WHEN	trace of recent activity of user – as noted, with ontoPIM if the last email read was from Tiziana and the user selects an Internet telephone service, then the telephone number would default to Tiziana's number.
WHO	profile/preferences of user and long-term activity – an example of the first, profile preferences, is when a post code is selected in Snip!t, we may link to a web-based route service to the selected post code from the user's normal address. For the latter, long-term activity, we should increase the likelihood of "John 3" being a bible verse if the user has previously clicked through bible-verse links.

Whilst these are all in the context of purely digital environments, it is clear that similar categories are common in ambient intelligence. For example, the user saying "hotter" is likely to mean something different when in a shower than in a kitchen (WHERE).

Notably missing from the above list is why – the inferences we draw from the other 4 Ws and how … the next section.

3.3 Architecture

The recogniser–service paradigm has proved very flexible; in particular by separating these rather than having recognition tied closely to a single action, it is easy to add additional actions for an already existing recognised type. Again this is also true for intelligent ubiquitous environments. Some proposed solutions have involved directly learning stimulus–response pairs, however having some form of more intelligible intermediate representation helps explanation and allows humans to fine tune the rules, thus making the rule set more robust as technology changes (e.g. a new kind of temperature sensor).

onCue recognisers were also recursive in the sense that the output of one recogniser may trigger another. Similarly in Snip!t, the syntactic bible verse is triggered by the lexicon lookup recogniser for chapter names and abbreviations. This seems a generally useful paradigm either with separated levels of recognition (fairly common in ubicomp) or a flexible service as in onCue.

Snip!t is currently restricted to one kind of data – web pages. In contrast onCue worked across applications … but only through clipboard use. The clipboard is usually the only truly application independent source of data on a GUI platform. Ideally onCue would have fitted more closely into applications, but this is hard without per-application coding. Interestingly Citrine, another recent application in the data-detector tradition, is based purely on intelligent clipboard to clipboard interactions, for exactly the same reasons [Stylos, 2004]. Similarly ubicomp environments need some level of standardisation and discovery so context-sensitive interactions can "listen in" to other devices.

Whilst onCue and ontoPIM are designed to operate on a single machine and serve a single user, Snip!t is web-based and designed to serve many users simultaneously. The current implementation is monolithic, but in order to scale future implementations we may need to separate out the recognition, services and user interaction management into separate web services. However, in order to have context sensitive recognition, either the recognition service has to send a relatively large number of context-independent suggestions for the user-interaction component to filter based on context, or the user interaction component has to send some representation of current context with request for recognition. As well as performance considerations, there are privacy and security implications if contextual information is passed between web services. These problems may be addressed by adding more interaction between the services : eg. a request for recognition might lead to a request for a specific type of context.

4. Summary

Our discussion of onCue, Snip!t and ontoPIM has allowed us to suggest and explore several more general issues for context-sensitive interaction. The issue of appropriate intelligence has been noted previously, but is particularly critical when the system is proactive or where it is costly to change 'intelligent' defaults. The simple breakdown of types of context is already proving useful in thinking about practical sources of useful contextual information. Also the different elements have different temporal properties, thus implying that activation-based context may need several flavours of activation. Finally we have described a few architectural issues that are critical for this kind of digital environment, but which may also (as with the other issues) be important for more physical ubicomp applications.

5. Acknowledgements

Supported by the Information Society Technologies (IST) Program of the European Commission as part of the DELOS Network of Excellence on Digital Libraries (Contract G038-507618). onCue was developed by the aQtive team, now in diaspora, including Alan's co-directors Russell Beale and Andy Wood. Snip!t was originally conceived following work by Jason Marshall on bookmarking behaviour [Dix, 2003] and has been developed with Fiona Dix and is hosted by Hiraeth Mixed Media.

6. Questions for the workshop

As always any research raises more questions than it answers! Two that are particularly 'hot' at present are:

(i) *separation of concerns*: how to represent context to make it extensible and how to deal with potential distribution, when context is not stored where it is used.

(ii) *reasoning*: how to 'reason' over context (e.g. activation models, fuzzy, etc.) and how to 'explain' contextual inferences to users (or more generally make them comprehensible).

7. Bio and team

The first author, Alan Dix is Professor of Computing at Lancaster University. He is author of one of the key texts in human–computer interaction and has interests in diverse aspects of HCI. His early interests in intelligent interfaces include intelligent hypertext in the late 1980s and Query-by-Browsing, automated database query inference, in the early 1990s. In 1998 with Russell Beale and Andy Wood, he founded a company, aQtive, focused on intelligent internet access, which sadly died in the dot.com collapse of 2000 ☹ The current work is in the context of the TIM sub-project of DELOS and involves a collaboration between the University of Rome La Sapienza, University of Athens and Lancaster University. The aim of TIM is to enable task-based interaction through an integrated 'filing system' organised using a personal ontology.

REFERENCES

[1] A. Dix, R. Beale and A. Wood (2000). Architectures to make Simple Visualisations using Simple Systems. *Proceedings of. AVI2000,* ACM Press, pp. 51–60.

[2] A. Dix (2002). beyond intention – pushing boundaries with incidental interaction. *Proceedings of Building Bridges: Interdisciplinary Context-Sensitive Computing*, Glasgow University, 9 Sept 2002. http://www.hcibook.com/alan/papers /beyond-intention-2002/

[3] A. Faaborg and H. Lieberman (2006). A Goal-Oriented Web Browser. *Proceedings of CHI 2006*. ACM Press. pp. 751–760

[4] V. Katifori, A. Poggi, M. Scannapieco, T. Catarci, and Y. Ioannidis (2005). OntoPIM: how to rely on a personal ontology for Personal Information Management. In *Proc. of the 1st Workshop on The Semantic Desktop*.

[5] W. Hall (1997). The History of the Microcosm Project. University of Southampton. http://www.mmrg.ecs.soton.ac.uk/projects /microcosm.html

[6] A. Dix and J. Marshall (2003). At the right time: when to sort web history and bookmarks. In *Volume 1 of Proceedings of HCI International 2003*. J. Jacko and C. Stephandis (ed.). Lawrence Erlbaum Associates, 2003. pp. 758–762

[7] B. Nardi, J. Miller, and D. Wright (1998). Collaborative, Programmable Intelligent Agents, *Communications of the ACM*, March 1998. **41**(3):96–104

[8] M. Pandit and S. Kalbag (1997). The selection recognition agent: Instant access to relevant information and operations. *Proceedings of Intelligent User Interfaces '97*. ACM Press. pp. 47–52

[9] J. Stylos, B. Myers and A. Faulring (2004). Citrine: providing intelligent copy-and-paste. *Proceedings of UIST'04*. ACM Press. pp. 185–188

[10] A. Wood, A. Dey and G. Abowd (1997). CyberDesk: Automated Integration of Desktop and Network Services. Technical Note in the *Proceedings of 1997 conference on Human Factors in Computing Systems* (CHI '97), pp. 552–553.