Tasks = data + action + context: automated task assistance through data-oriented analysis

Alan Dix

Computing Department, InfoLab21, Lancaster University
Lancaster, LA1 4WA, UK
alan@hcibook.com
http://www.hcibook.com/alan/papers/EIS-Tamodia2008/

Abstract. Human activity unfolds partly through planning and learnt sequences of actions, and partly through reaction to the physical objects and digital data in the environment. This paper describes various techniques related to automatic task assistance that take this role of data as central. Although this brings additional complexity, it also offers ways to simplify or bypass problems in task inference that otherwise appear difficult or impossible. Although the focus in this paper is on automated task support, the importance of objects and data in understanding tasks is one that applies to other forms of task analysis in the design process.

Keywords: task inference, data detectors, automated task support, intelligent user interfaces, task as grammar,

1 Introduction

One morning recently, whilst having breakfast, I served a bowl of grapefruit segments and then went to make my tea. While making the tea I went to the fridge to get a pint of milk, but after getting the milk from the fridge I only just stopped myself in time as I was about to pour the milk onto my grapefruit! I am sure everyone reading this has made a similar mistake, but it is not just an amusing anecdote; the analysis of such mistakes is the grist of human error analysis and equally tells us critical things about even error-free tasks.

Note that not all mistakes are equally likely: I would be unlikely to pour the milk onto the bare kitchen worktop or onto a plate of bacon and eggs. This is a form of capture error: the bowl containing the grapefruit might on other occasions hold cornflakes; when that is the case and I am standing in the kitchen with milk in my hand, having just got it out of the fridge, it would be quite appropriate to pour the milk into the bowl.

In all but the most repetitive, routinised, or organisationally prescribed settings, the actual evolution of the goal into human activity is far more complex and situated then we can easily capture in simple task-hierarchies and plans. Real activity involves procedural, reactive and consciously considered actions and for over 20 years, many in HCI have argued that the complexity of 'situated action' renders more formal task

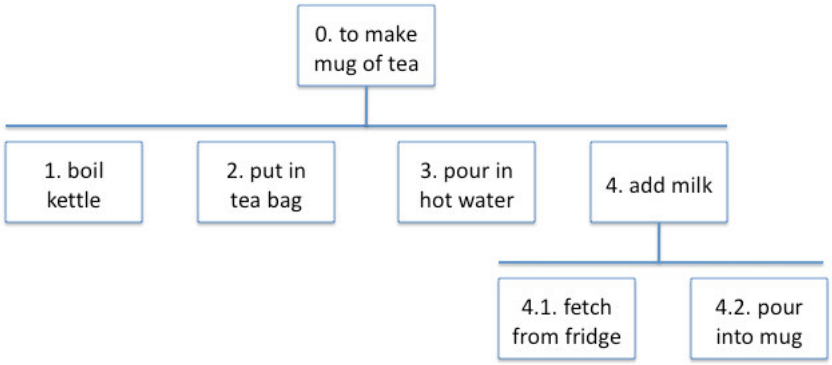

Figure 1. Making a mug of tea (sorry no teapot)

or goal analysis incorrect, obsolete and irrelevant. In contrast, in my keynote at the first Tamodia in 2002, I discussed how some of these more contextual or situational elements could be drawn into a formal task model: the role of information, other people, physical artefacts, triggers for action, and placeholders keeping track of where we are in a task [1].

This subtle complexity of real human activity is difficult for a human analyst to adequately describe in more formal terms; even when that formalisation is properly understood to be partial and provisional. However, this is far worse when the 'analyst' is a machine! As a human analyst we can ask what is going on in a user's head, but automated analysis typically has only the trace of user actions available and no real understanding of human activity and purpose.

Over the years I have intermittently worked on aspects of task inference and automated support of user activity. In this paper I will reflect on the relation of this to more human task analysis and the way they inform one another. Many of the techniques described are being worked on together with colleagues at Lancaster, University of Rome "La Sapienza", University of Athens, University of Peloponnese and Universidad Autonoma de Madrid. I will mention other names explicitly in the paper, but in other places when I say "we", this refers to joint work with various colleagues in this group.

2. The Best Laid Plans and Reactions

We'll start by looking in a little more detail at the milk in the grapefruit error. Figure 1 shows a HTA of the task of making a mug of tea. It is subtask 4 that is interesting. It has two further subtasks and one would normally write a plan such as:

Plan 4.
    if milk not out do 4.1
    then do 4.2

However, maybe it is more like:

Plan 4.
    if milk not out do 4.1
    when milk in hand do 4.2

The capture error is then understandable as the "eat some cereal" task will also have a plan with something like "when milk is in hand pour into bowl".

The first version of the plan is really a 'planned' plan, or maybe a proceduralised one, where the sequence of actions is in some way explicitly or implicitly remembered. However, the second is effectively a stimulus–response reaction based on conditions in the environment – the sequenced and hierarchical structure will still be there, but are maintained because it unfolds as the human actions interact with the environment, not because the order is remembered. Furthermore, the user may even be performing some form of explicit means-end analysis "in order to add milk I need a bottle of milk", … "in order to get a bottle of milk I need to open the fridge".

We can arrange these types of sequenced activity by whether they are explicit or implicit and whether they are pre-planned or environment-driven:

	pre-planned	environment-driven
explicit	(a) following known plan of action	(b) means–end analysis
implicit	(c) proceduralised or routine actions	(d) stimulus–response reaction

From watching a user we often cannot tell which of these is the reason for a particular sequence of observed actions. While these are very different in terms of cognitive activity, they are virtually indistinguishable from behaviour alone.

Now we might assume that a well-practiced user will learn frequently repeated tasks: that is, even if the user starts off with an explicit plan (a) or are means-end driven (b), they will eventually end of with proceduralised or routine actions (c) – practice makes perfect. Certainly this is true of repeated actions in sports and music.

However, theorists advocating strong ideas of the embodied mind would argue that we are creatures fitted most well to a perception–action cycle and where possible are parsimonious with mental representations allowing the environment to encode as much as possible.

> "*In general evolved creatures will neither store nor process information in costly ways when they can use the structure of the environment and their operations on it as a convenient stand-in for the information-processing operations concerned.*" ([2] as quoted in [3])

Clark calls this the "007 principle" as it can be summarised as: "*know only as much as you need to know to get the job done*" [3].

In the natural world this means, for example, that we do not need to remember what the weather is like now as we can feel the wind on our cheeks or the rain on our hands. In a more complex setting this can include changes made to the world (e.g. the bowl on the worktop) and even changes made precisely for the reason of offloading information processing or memory (e.g. ticking off the shopping list). Indeed this is one of the main foci of distributed cognition accounts of activity [4].

It is not necessary to take a strong embodied mind or even distributed cognition viewpoint to see that this parsimony is a normal aspect of human behaviour – why bother to remember the precise order of doing things to make my mug of tea when it is obvious what to do when I have milk in my hand and black tea in the mug?

Of curse parsimony of internal representation does *not* mean *no* internal representation. The story of the grapefruit bowl would be less amusing of it happened all the time. While eating breakfast it is not unusual for me to have both a grapefruit bowl and a mug of tea out at the same time – so why don't I make the same mistake every morning? In fact I do have some idea of what follows what (plan) and also have some idea that I am "in the middle of making my tea" (context, schema).

Together these factors: environment, plans, context inform the actions we perform in the world.

We will look at each of these factors and how they impact automatic inference and support of users' tasks.

3. Environment – data driven interaction

In ubiquitous computing, the instrumentation of the environment is a major issue in itself and so inferring user behaviour from the environment is very difficult. In contrast, in the purely digital world of the desktop or web, we have, in principle, relatively easy access to the complete digital environment. This makes certain forms of data-driven interaction particularly easy.

One form of this are "data detectors", which usually use some form of textual analysis to identify potential key terms, dates, etc. in small bodies of text such as email messages, or your current selection. The initial work on data detectors occurred in the late 1990's when there were a number of other data detector projects at Intel (SRA) [5], Apple [6] and Georgia Tech (CyberDesk) [7]. The Apple work led to the inclusion of Apple Data Detectors in the operating systems (and still there albeit often unused). When activated (and when using a compliant application) small contextual menus appear over selected words/phrases in the text of the current document or email.

At around the same time I was involved in the development of onCue [8], a small "intelligent" toolbar. This sat at the side of the screen and watched for changes to the clipboard (through copy-paste); when the clipboard changed, 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 internal architecture of onCue consisted of two main kinds of components, recognisers and services, linked by a blackboard-like infrastructure. The recognisers examined the clipboard contents to see if they were a recognised type (post-code, 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. This separation (itself building on CyberDesk [7]) was an important difference from most other systems where the two were linked as it meant that services could easily be added for previously recognised types adding to the potential for third party additions (through small XML "Qbits").

OnCue used the clipboard as its focus as the clipboard is usually the only truly application-independent source of data on a GUI plat- form. Ideally onCue would have fitted more closely into applications, but this is hard without per-application coding, even Apple found this despite controlling the platform! For exactly the same reasons, Citrine, another recent application in the data-detector tradition, is based purely on intelligent clipboard-to-clipboard interactions, offering intelligent transformations between types of clipboard content [9].

Citrine is part of a recent small resurgence in work on data detectors, including headlines a few years ago due to disputes about Microsoft SmartTags. Yahoo! also have ways for web developers to include context-sensitive searches into their web pages keyed on phrases in the page contents, and Amazon have recently announced a similar mechanism to link to books and other products. More interesting compared with these more hand-crafted links is the CREO system [10]. CREO takes several large ontologies of general knowledge and uses these to build indices of critical words and phrases. As the user browses the web a plug-in looks for matching words in the web pages visited and adds contextual links to web based interactions concerning the topic of the words. The information for this is also used to allow the user to train the systems to do new actions by example.

The mode of operation of CREO is reminiscent of an older body of work that started over 15 years earlier in the HyperText community, where notions of external linkage were important. Microcosm [11] developed at Southampton pioneered the use of automatic links. This used an index of key terms attached to a particular content. When the user viewed a document any key terms present in the index became live links in the document. Note that, with the exception of CREO, most of the data detectors, including onCue, rely on largely syntactic/lexical matching using regular expressions or other patterns whereas Microcosm was lexicon based.

Snip!t (www.snipit.org) is a web-based system allowing users to bookmark sections of a web page rather than just the URL of the page itself [12]. It has been developed intermittently over a period of about 5 years based on initial user studies of bookmarking showed that users want to be able to recall a portion of a page [13]. This is now more common in tools such as Google Notebook and various annotation services such as Bricks [14] and MADCOW [15]. Snip!t inherits the recogniser–service architectire of onCue but running server-side rather than on a user's own machine. This allows it to access larger data sources, like Citrine and Microcosm, and so it performs a mixture of lexicon look-up and syntactic recognizing of suitable types, including hybrids. For example if a word matches one of the common names from a US census dataset of first names, it triggers a full syntactic analysis to check whether the surrounding text is really a name.

In many ways these data detector and related services are very much like a butler who, seeing you in the kitchen holding a pint of milk and a mug of black tea, says "would you like me to pour milk in your tea, sir?" However, the above systems all have little if any adaptation to the user, so are like an absent-minded butler who always asks the same even though you never take milk in your tea.

In order to make this kind of data-detector more individual, in the TIM project we are connecting this data detector technology with a personal ontology [16]. A personal ontology is an explicit store of personal information such as friends, work colleagues, projects, papers, and addresses, including the connections between them. There are various usability issues relating to how one encourages a user to produce and maintain such and ontology, but in general the process will be semi-automatic. The GNOWSIS project [17] has found that by mining very explicit desktop data such as address books, email messages etc. it is possible to build at least part of the data we would want to see in such an ontology.

If one assumes that such a personal ontology exists, then the terms in the ontology can be matched in text alongside public information sources such as gazetteers. So as well as recognising that Pisa is a City, it will also recognise that "Fabio" is the first name of an academic in my personal ontology and then be able to suggest things that are appropriate for an academic such as looking him up in DBLP.

4. Context – what to do and what to do it to

Of course a human aide would not only know about you as an individual (such as whether you like milk in your tea), but also know something about what is happening to you now (such as making tea or making pancakes).

Context recognition and context awareness have become especially important in ubiquitous and mobile computing where interactions with the world are central, but also in purely digital domains such as adaptive hypertext, and e-learning. It may be useful even in purely digital settings to know, for example, whether the user is stressed or relaxed, with colleagues or on her own. However, this paper will focus on context that can be inferred from the digital domain itself.

To do this, we take the personal ontology and then use spreading activation in order to represent what are the 'hot spots' in the ontology at any particular moment [18]. Spreading activation has its roots in cognitive psychology [19] and so has the potential to model context in a way somewhat resembling a human. The basic idea is that when an event or document refers to some entity in the ontology it becomes 'activated' (say I have an email from Vivi, then the entity representing Vivi gets an initial high activation). The algorithm then 'spreads' the activation by making entities connected to Vivi a little activated, then those connected to these slightly less active entities. There are problems, such as loops in the ontology, which can set up self-reinforcing feedback, but these can be controlled with care in the detailed algorithms.

Now imagine I receive a second email that mentions "George". I may know several people called George, so on its own any form of digital assistant can at best suggest it is one of a long list. However, with the spreading activation, the George who is part of the same project and in the same country as Vivi will be 'hotter; than others and so be the first suggestion. Also if the next action I am performing requires

a city name, then an assistant can pre-complete the form with "Athens" as a suggestion as this is the city that is most highly activated.

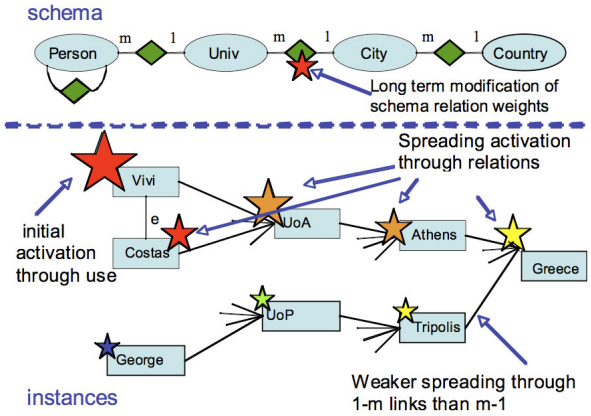

Figure 2. Spreading activation through a personal ontology

A related approach is being used to create declarative representations of the relationship between web form fields [20]. Whenever a user enters test into form fields an automated system records the contents and then attempts to infer the relationship between the fields. If the fields are completely unrecognised it can do nothing (although this would be an appropriate point to suggest that the users store the data in their personal ontology!) However, if several form fields are found in the personal ontology, then an algorithm searches for 'best' paths between the fields.

There are typically several such paths hence the need for weighting. For example "Lancaster University" is related to "Alan Dix" by being his institution, but also the institution of Devina his work collegue. That is we have two possible paths:

(a) name_of / Person / member / Institution / has_name

(b) name_of / Person / colleague / Person / member / Institution / has_name

The system would give the first of these a higher weight because it is 'shorter' based on number of relationships traversed and their branching factors. Potentially, this could also use the current activation to weight more highly paths through 'hot' entities.

Next time the user comes to the form, the system knows not only what data was entered before (as in standard browser auto-completion), but also the relationship between them in abstract terms. So if the user enters Antonella into the first field, the system traverses path (a) and auto-completes the second field not with "Lancaster University" (the last value entered), but with "University of Rome" as that is the name of the institution that Antonella is a member of.

5. Sequence – from traces to plans

Performing a task leads to some observable trace of actions. This trace of real activity is often the meeting point of different views of the world. Even if we disagree on interpretations of events, we can often (although not always!) agree on what actually happened. For this reason, in earlier work, I have referred to traces as a "ubiquitous semantics" for different user interface formalisms from task analysis to system models [21].

One way to view an HTA is as a grammar over this trace of actions. Personally I have found this a useful way to teach about task analysis, and have included this in the teaching materials for the Human–Computer Interaction textbook (although not yet in the actual text) [22,23]. As an illustration, figure 3 shows a simple HTA of cleaning a house and figure 4 shows how this can be used to build a 'parse tree' of a trace of actual actions (trace on the left, parse tree on the right). Note that unlike a textual grammar, the task grammar includes interleaved activities (the instance of task 4 "empty the dust bag" in the middle of the execution of task 3 "clean the rooms").

This can be applied to practical task analysis. In his thesis work, Stavros Asimakopoulos has used the "HTA as grammar" approach to supply chain forecasting [24]. Interviews with system developers and forecasters included accounts of actual forecasting activity (for the developers, envisaged; for the forecasters, from experience). These (partial) activity sequences were then matched against a normative task analysis based on the literature allowing an analysis of discrepancies between normative and actual tasks.

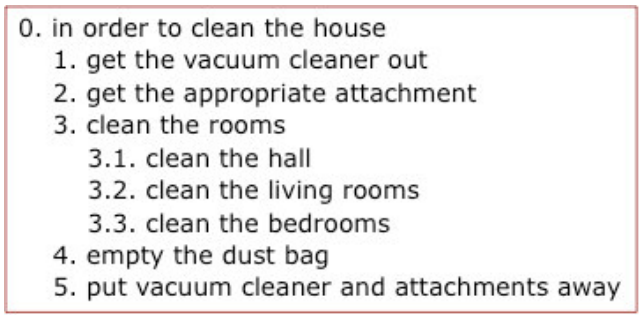

Figure 3. HTA for cleaning a house (from [22]

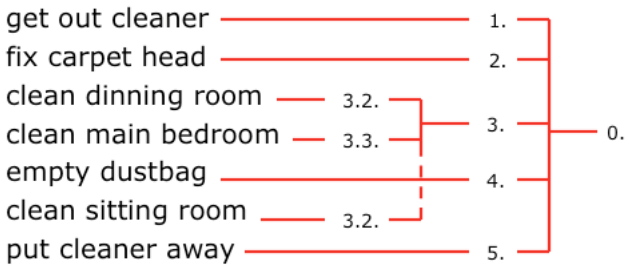

Figure 4. Parsing a trace using an HTA (from [23] and [24])

This approach can be used inductively too, and indeed direct observation is one of the normal sources for task analysis. As an analyst one is looking at a sequence of actual actions and attempting to infer a hierarchical (or other) structure on those actions. To do this the analyst uses a combination of common sense, domain knowledge and interaction with users in order to ascertain that, for example, putting money in a slot is part of parking a car.

For automated analysis this becomes far more complicated – I have been told that the general problem of inferring a hierarchical grammar from a sequence is computationally hard (either NP or at least n^k for some large 'k'!). However, in practice things are not quite as bad as this suggests. Indeed various forms of action/tasks inference have been common in the literature with the heyday in the early 1990s. The most well known example is Allan Cypher's Eager [25], but there have been many such systems using various algorithms including neural networks and hidden Markov models [26,27,28]. In recent years certain (albeit limited) forms of task inference can be found in commercial systems such as auto-completion of lists in Microsoft Office or form auto-fill features in web browsers … but the former emphasises the need to put any such 'intelligent' features within an appropriate interaction framework. (See "appropriate intelligence" in [29], especially the principle that one should design foremost for the times when the intelligence, inevitably, fails and make interaction graceful at such times.).

In some cases data-focused interactions can give rise to emergent task sequencing: if the output of a basic user action is some form of data, then this becomes the locus for the next action, etc. However, data linkages can also be used to make the job of inferring structure from tasks sequences easier.

One of the problems in inferring task structure from traces of user activity is that we interleave different tasks. I may be writing a paper, but occasionally reading or writing an email while I do so, or maybe taking a break to play a game of solitaire. Email reading on its own is perhaps one of the most challenging domains as perforce the mails arriving are related to different higher-level tasks and yet they typically get read in arrival order not a task at a time.

This is similar to the case in the kitchen where I maybe alternating between making tea, serving grapefruit and chatting to my wife. Although the milk is somewhat problematic, it is obvious that boiling the kettle is connected with making the tea because the water from the kettle goes into the mug not the grapefruit bowl. That is

the shared physical objects in the environment can be used to establish links between low-level actions.

We can do the same thing in the digital domain. If we keep track of what digital objects are produced or used by different user actions then this creates a linkage between them. For example, if I copy a date from an event in my calendar and paste it into a hotel booking form, I create an implicit link between the two actions.

If the user types the data, things become more difficult. For example, if the result of a search produced "Miguel" and then I typed "Madrid" into a text box. However, here the algorithm described at the end of the previous section again comes into play and offers a way to establish potential relationships through the personal ontology.

Now, assuming we have these data links between low-level actions, we can start to 'pull out the threads' of tasks from the undifferentiated interleaved sequence of actions. This is a bit like finding one end of a string of pearls in a jewellery box and gently pulling the whole string (see Fig. 5). In principle these data links could take place days, weeks or months later and still be detectable.

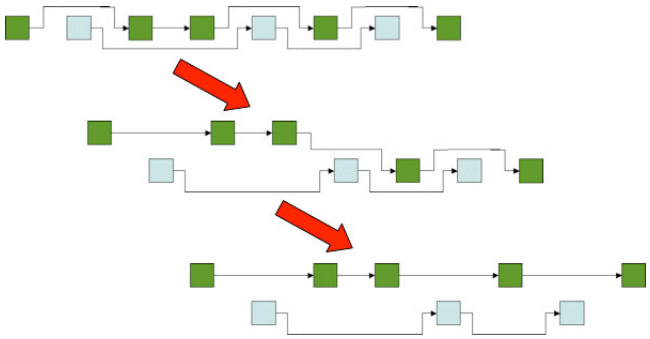

Figure 5. Pulling out task threads form interleaved user actions

Once this thread has been pulled out we have a task sequence that is not confused by interleaved activities of other kinds and thus far more amenable to further analysis. For example, if a sequence of low-level actions A, B, C is detected and action A is later performed then the option of performing actions B ad C can be proposed. Furthermore the fact that we have the data link between them means we can auto-complete the parameters of the subsequent actions. Of course, given this sequence, some of the more sophisticated inference techniques in the programming by example / by demonstration literature can be used [28]. The crucial thing is that data linkage turns what seems like a near impossible problem into a relatively simple one.

Of course nothing is as trivial as that and there are some complications. The task thread is a data flow and so may not be a simple sequence, but instead DAG (directed acyclic graph) with time-based total ordering. Also any inferred data linkages mean that there is a level of uncertainty associated with the detection of threads, leading to several potential task threads with some level of confidence associated with each.

In principle it would also be possible to infer a level of hierarchical structure, either through the temporal structure, by looking for common sub-sequences of actions; or through the data structure, by looking at braches in the DAG. However, this seems an

appropriate level to rely more strongly on the user. When a sequence of actions is suggested an option can be "name it". As this the point, the task sequence is being used, and so is an appropriate moment to request small (but optional) additional user effort. If the user does this, it means that (i) the user has effectively agreed that these actions form a meaningful task chunk which can then be treated as atomic in further inference and (ii) the chunk has a meaningful name that can be used in future suggestions, or even to share with others.

6. Discussion

A key theme in this paper has been the interplay between data and action. Taking this seriously allows us to consider various forms of automated task support that would otherwise seem difficult or impossible. This is *not* to say that we should adopta purely data-oriented view, but by that using data-focused analysis alongside ways to capture or inference more sequential or structured plans, we both create more robust inference and make the detection of structure easier. All of this is set within the context of interaction, some of which we can attempt to infer, and some, such as the deeper intentions of the user, we need to defer to the user's own decisions and control. Indeed, as we saw especially in the final discussion of the preceding section, we are likely to obtain more reliable results if we consider an ongoing dialogue of suggestion and observation rather than a more 'waterfall' approach of observe, infer then automate. Furthermore, such inference processes could easily operate symbiotically alongside more user-initiated scripting such as Apple Automator or Yahoo! Pipes, further increasing user control whilst still offering rich assistance.

With the exception of task threading, the work described in this paper is mostly implemented, but as distinct units, and we are working on bringing this together, within a unified architecture. For various reasons intelligent and adaptive interfaces, whilst continuing to have their strong advocates, got a bad name in the general HCI community for many years. Some of this was due to factors that still need to be treated with care: inappropriate choice of algorithms; the prevailing "user in control" ethos of direct manipulation; and detailed design issues, not least the lack of 'appropriate intelligence' in that often software is designed well for the test cases where it produced good results, but copes less well when the results are less clear. However, some of the problems were simply due to the limited computational power available 15 years ago – intelligent algorithms are typically expensive algorithms. With each PC one thousand times more powerful than during this early blossoming, and the raw computational power in the internet rivalling a (single) human brain, the times seem pregnant for more automated (but not autocratic!) assistance.

While the focus of the work described here is automated task assistance, broader lessons for human analysis and task design also emerge. We started with a non-automated, non-digital example of tea, milk and grapefruit. The focus on artefacts and physical objects as part of the task is central to understanding the errors that occur; and of course are also valuable for re-designing tasks and environments to prevent those errors occurring. The artefact-focus has also proved very powerful in uncovering long-term or complex tasks in the non-digital world [30]. Back in my keynote at the first Tamodia, I emphasised the importance of explicitly including

artefacts and environment (both physical and digital) within task analysis. In this paper I have principally shown how taking into account the world of data can help the computer to predict, suggest and automate aspects of user tasks. If this can help the computer, it can help people. While there are exceptions (e.g. [31,32]), many forms of task analysis still portray the user's plans as largely pre-ordained and un-reactive … effectively a disembodied user thinking and acting without recourse to the world. If the role of artefacts and data is fully represented in task analysis then this could lead to better systems designs that make available prompts and external resources to users; so that users' own choices and actions become easier, less cognitively taxing and less error prone: designing for the embodied user.

7. Acknowledgements

This paper is drawing on ongoing collaborative work with a number of people including: Tiziana Catarci, Yannis Ioannidis, Azrina Kamaruddin, Akrivi Katifori, Giorgos Lepouras, Nazihah Md.Aki, Estefanía Martín, Miguel Mora, Antonella Poggi, Devina Ramduny-Ellis, and Costas Vassilakis. It was also supported by the EU funded DELOS Network of Excellence on Digital Libraries.

For links to related work please see:

http://www.hcibook.com/alan/papers/ EIS-Tamodia2008/

References

1. Dix, A.: Managing the Ecology of Interaction. In Pribeanu, C., Vanderdonckt, J. (eds.): Proceedings of Tamodia 2002 - First International Workshop on Task Models and User Interface Design, pp. 1--9. INFOREC Publishing House, Bucharest (2002)
2. Clark, A.: Microcognition,: Philosophy, Cognitive Science and Parallel Processing. MIT Press, Cambridge, MA (1989)
3. Clark, A.: Being There: Putting Brain, Body and the World Together Again. MIT Press, Cambridge, MA (1998)
4. Hollan, J., Hutchins, E., Kirsh., D.: Distributed Cognition: Towards a New Foundation for Human–Computer Interaction Research. Chapter 4 In: Carroll, J. (ed.) Human–Computer Interaction in the New Millennium. pp. 75–94. Addison-Wesley Professional, Boston (2002).
5. Pandit M., Kalbag, S.: The selection recognition agent: Instant access to relevant information and operations. In: Proc. of Intelligent User Interfaces (IUI 97), pp. 47–52 ACM Press (1997)
6. Nardi, B., Miller, J., Wright, D.: Collaborative, Programmable Intelligent Agents. Communications of the ACM, 41(3), 96–104 (1998)
7 Wood, A., Dey, A., Abowd, G.: Cyberdesk: Automated Integration of Desktop and Network Services. In: Proc. of the Conference on Human Factors in Computing Systems (CHI 97), pp. 552–553, ACM Press (1997)
8. Dix, A., Beale, R., Wood, A.: Architectures to make Simple Visualisations using Simple Systems. In: Proc. of. Advanced Visual Interfaces (AVI 2000), pp. 51–60, ACM Press (2000)

9. Stylos, J., Myers, B., Faulring, A.: Citrine: providing intelligent copy-and-paste. In: Proc. of the 17th Symposium on User Interface Software and Technology (UIST 2004), pp. 185–188, ACM Press, (2004)
10. Faaborg, A., Lieberman, H.: A Goal-Oriented Web Browser. In: Proc. of the Conference on Human Factors in Computing Systems (CHI 2006), pp. 751–760, ACM Press (2006)
11. Hall, W., Davis, H., Hutchings, G..: Rethinking Hypermedia: The Microcosm Approach. Kluwer Academic Publishers, Norwell, MA, (1996)
12. Dix, A., Catarci, T., Habegger, B., Ioannidis, Y., Kamaruddin, A. Katifori, A., Lepouras, G., Poggi, A., Ramduny-Ellis, D.: Intelligent context-sensitive interactions on desktop and the web. In: Proceedings of the international Workshop in Conjunction with AVI 2006 on Context in Advanced Interfaces. pp. 23–27, ACM Press (2006)
13. Dix, A., Marshall, J.:At the right time: when to sort web history and bookmarks. In: Proc. of HCI International 2003, vol 1, pp. 758–762 (2003)
14. Haslhofer, B. Hecht, R.: Joining the BRICKS Network - A Piece of Cake. In: The International EVA Conference (2005)
15. Bottoni, P., Civica, R., Levialdi, S., Orso, L., Panizzi, E., and Trinchese, R. 2004. MADCOW: a multimedia digital annotation system. In Proceedings of the Working Conference on Advanced Visual interfaces (Gallipoli, Italy, May 25 - 28, 2004). AVI '04. ACM, New York, NY, 55-62.
16. Katifori, A., Vassilakis, C., Daradimos, I., Lepouras, G., Ioannidis, Y., Dix, A., Poggi, A., Catarci, T.: Personal Ontology Creation and Visualization for a Personal Interaction Management System. In: Workshop on The Disappearing Desktop: Personal Information Management 2008. CHI2008, (2008)
17. Sauermann, L.: The Gnowsis Semantic Desktop for Information Integration. In: The 3rd Conference on Professional Knowledge Management, pp. 39–42 (2005)
18. Katifori, A., Vassilakis, C., Dix, A.: Using Spreading Activation through Ontologies to Support Personal Information Management. In: Common Sense Knowledge and Goal-Oriented Interfaces (CSKGOI 2008) (workshop at 2008 International Conference on Intelligent User Interfaces (IUI 2008). CEUR Workshop Proceedings Vol 323 (2008).
19. Anderson, J.: A spreading activation theory of memory. Journal of Verbal Learning and Verbal Behaviour, 22, 261–295 (1983)
20. Dix, A., Katifori, A., Poggi, A., Catarci, T., Ioannidis, Y., Lepouras, G., Mora, M.: From Information to Interaction: in Pursuit of Task-centred Information Management. In: DELOS Conference 2007 (2007)
21. Dix., A.: Towards a Ubiquitous Semantics of Interaction: phenomenology, scenarios and traces. Interactive Systems. In Forbrig, P., Limbourg, Q., Urban, B., Vanderdonckt, J. (eds.): DSV-IS 2002. LNCS, vol. 2545, pp. 238–252. Springer, Heidelberg (2002)
22. Dix, A., Finlay, J., Abowd, G., Beale, R.: Human-Computer Interaction, third edition. Prentice Hall, Harlow (2004)
23. Dix, A., Finlay, J., Abowd, G., Beale, R.: Chaper 15 sides. Online Teaching Resources for Human-Computer Interaction. (2004) http://www.hcibook.com/e3/resources/
24. Asimakopoulos, S., Fildes, R., Dix, A.: Grammatically interpreted task analysis for supply chain forecasting. In: Proceedings of the 10th British HCI Conference, vol. 2, pp. 235-237. British Computer Society (2005)
25. Cypher. A.: Eager: Programming repetitive tasks by example. In: Proc. of the Conference on Human Factors in Computing Systems (CHI 1991). pp. 33–39, ACM Press (1991)
26. Finlay, J. Beale. R.: Neural networks and pattern recognition in human-computer interaction. ACM SIGCHI Bulletin, 25(2). 25–35 (1993)
27. Dix, A., Finlay, J. Beale. R.: Analysis of user behaviour as time series. In Monk, A., Diaper, D., Harrison, M. (eds): Proceedings of HCI'92: People and Computers VII. pp. 429–444, Cambridge University Press (1992).

28. Lieberman, H.: Your wish is my command: programming by example. Morgan Kaufmann, San Francisco (2001)
29. Dix, A., Beale, R., Wood, A.: Architectures to make simple visualisations using simple systems. In Proceedings of the Working Conference on Advanced Visual interfaces, AVI '00. pp. 51–60, ACM, New York, (2000)
30. Ramduny-Ellis, D., Dix, A., Rayson, P., Onditi, V., Sommerville, I., Ransom, J.: Artefacts as designed, Artefacts as used: resources for uncovering activity dynamics. In Jones, P., Chisalita, C., van der Veer, G. (eds.): Special Issue on Collaboration in Context: Cognitive and Organizational Artefacts, Cognition, Technology and Work, 7(2). pp: 76–87 (2005).
31. Task Analysis Through Cognitive Archeology Frank Spillers. In Diaper, D., Stanton, N. (eds): The Handbook of Task Analysis for Human-Computer Interaction. pp. 279–290. Lawrence Erlbaum Associates, 2004
32. Dix, A., Ramduny-Ellis, D., Wilkinson, J.: Trigger Analysis: Understanding Broken Tasks. In Diaper, D., Stanton, N. (eds): The Handbook of Task Analysis for Human-Computer Interaction. pp. 381–400. Lawrence Erlbaum Associates, 2004