2011

Use Centric HCI Requirements Elicitation

Robert B. K. Brown
University of Wollongong, bobbrown@uow.edu.au

Use Centric HCI Requirements Elicitation

Robert B. K. Brown
School of Information Systems and Technology
University of Wollongong
Australia
Email: bobbrown@uow.edu.au

Abstract

Among persistent issues in HCI are its separation from the other disciplines involved in the construction of Information Systems, any coherently extensive methodology and straight forward techniques for clustering interface elements for best effect. An approach to HCI which encompasses the full context of use (rather than just of usage) and entails more traditional SA&D within a single framework could be of some value. This paper introduces the HCI values of the Activity Theoretic System Architecture (ATSA) Method.

Keywords

Use Centrism, Activity Theory, HCI

Introduction

Often lauded as having great potential, Activity Theory (AT) may be on the cusp of offering not only some workable solutions to persistent issues in the design of Human Computer Interfaces (HCIs), but may also offer some reconciliation between the fields of Systems Analysis and design (SA&D) and HCI.

Moving beyond user orientation to "use" orientation (as represented by Activities in AT), that is, concentrating of what needs to be done by users, allows for an early-phase requirements elicitation method that can also inform some aspects of HCI design. The identification of tasks and "doings" on a work-station by work-station basis identifies which interface elements are requires by which users in which temporal contexts.

This paper briefly outlines how the Activity Theoretic System Architecture (ATSA) method may offer some new tools towards HCI specification.

Background: Problems with HCI

Concentrating largely on human computer interactions and system interfaces, Platt (2007) made a deliberately irreverent case for ascribing widespread failure in software to ignorance and arrogance among designers, who presume to know better than their users what they would need to have built. Gruden (1989) argued convincingly against a uniformity of UIs, or 'consistency' as he called it, declaring the notion to be 'unworkable'. Seemingly at odds with principles often held as in high regard in the HCI community, Gruden is actually calling for bespoke or contextual design (Beyer and Holtxblatt 1998). Gruden was quite comfortable with what he called 'internal consistency' but found no need to enforce any 'external consistency' between systems. The sense seemed to be that no universal answers were possible, so each UI should be 'fitted' to its system and its users' work environment. User centricity informed Nielsen (1994) when he codified ten heuristics for usability. Nielsen perceived that whilst, after Gruden, systems are not at all universal, the general cognitive behaviour of users however, may almost be.

Crucial among the difficulties hindering the design of interfaces which Myers (1993) identified were that designers could not easily understand the tasks conducted by users, and that existing theories and guidelines (at least as they were in 1993) were not up to the task. Myers cited a 1986 report by Smith which identified and detailed as many as 944 different guidelines. The evidence was that the quality of interfaces was attributable more to the skill of the designer, than to any particular method or guideline they may have employed. Myers further declared that though all HCI guidelines (as at 1993) recommended iterative design, it was particularly difficult to conduct, because there were no clear metrics for assessing what needed to be fixed between iterations, nor if an applied change would address the issue(s).

The implementation (coding) of UIs is made difficult, according to Myers, because (among other factors), they must be programmed from the "inside-out". Myers also found that the testing of interface software was difficult

(due to the qualitative nature of success-failure measures), that the tools to assist in interface implementation were overly complex and that programmer's reported difficulty with modular UI softwares.

Indicating perhaps a growing divide between those computer scientists who dabbled in interfaces and the then growing sub-discipline of HCI, Thimbleby (1990) deemed UI to be a portal to a computer system's functionality. He proposed a markedly more formal approach to the design of an interface but was careful to retain a statement of value for the creativity, thoughtfulness & effectiveness of a good UI. Offering architectural abstraction as a starting point for UI design, Torres (2002) set the design and development of UIs firmly within a larger development cycle (echoing the over-arching SDLC). Shelly and Rosenblatt (2011) went so far as to declare system output and UI design as the very first task in the design phase of the SDLC.

In Preece, Rogers and Sharp (2002) interaction design is portrayed with an architectural metaphor. The UI was a place where human conduct their business and may access a servile computerised assistant. For Preece, it is a dialogue about spaces and human-to-human transactions. Accepting the need to tailor a UI to the tasks of the user (client) group, Rosson and Carroll (2002) suggested the use of scenario based design, analysing the activities of the user by the use of tools such as the Hierarchic Task Analysis (HTA). Rosson and Carroll however make no mention of Activity Theory (AT) however, despite Bødker's declarations of 1991; evidence supporting the notion that (as Myers (1993) had bemoaned nine years earlier) existing theories and guidelines were simply not up to the task of designing UIs, and there was room for one to be developed or adopted.

Dix, Finlay, Abowd and Beale (2004) produced an authoritative and encyclopaedic teaching text, widely used for teaching HCI and UI design, which champions the work of Nielsen (1994) as establishing the groundwork for a pure HCI discipline. Whilst the text presents perhaps the most convenient and accessible 'one-stop-shop' for UI design tools, techniques and notions (largely of the type later labelled 'classics' by Lauesen (2005)) it should be noted that none of the tools they detail can operate from end-to-end of the HCI:UI lifecycle. Certainly none of them link or translate seamlessly (if, indeed, at all) into the broader system design lifecycle (SDLC) which (despite an increasing rhetoric of isolationism by HCI practitioners), still existed beside, if not around, the UI's own lifecycle.

Lauesen (2005) called for some kind of integrated expertise and a new mode of practice. Lauesen attempts to present a comprehensive review of HCI approaches, aiming to bridge the "two worlds" of programming and HCI. He was all too aware that they each held themselves aloof as the premiere craft.

Shneiderman and Plaisant (2005) commented on the diversity of the HCI field, listing such disciplines as sociology, anthropology, policy making and management. They sought to optimize an interface for speed of use, on the assumption that time was a user's primary consumable and they counselled designs which optimised the use of that finite resource. Their second priority was the avoidance of errors. As almost an afterthought, they endeavoured to reduce user frustration. For Shneiderman and Plaisant designing a UI was about tuning it to meet 'genuine human needs' (if not actual satisfaction) thus, moving beyond what they called the 'vague notion of user friendliness.'

Shneiderman and Plaisant required an initial elicitation of user needs (broken into tasks and sub-tasks), though quite how this is achieved remained as indistinct as it had ever been. In the second step, the designer should establish technical reliability, thus winning user trust. The third step enforced standardisation, integration, consistency and portability and implementation must be conducted under the best practices of project management.

Benyon, Turner and Turner (2005) echoed Preece in speaking of UIs in architectural metaphors and of the 'information space.' Benyon's HCI is entirely human centred; accessible, usable and engaging. Above all it had to be enjoyable to use. This is quite opposite to the more mechanical design philosophy of Shneiderman and Plaisant (2005) in the same year.

Bødker's landmark PhD thesis and text "Through the Interface" (Bødker 1991) and later, Nardi's text "Context and Consciousness: AT and Human-Computer Interaction" (Nardi 1995) laid out Activity Theory (AT) as a useful tool and theoretical framework for HCI study. Several proposals have come to light, notably the checklist idea (Kaptelinin 1999), however it has been stated that HCI has yet to benefit directly from AT (Vrazalic 2004).

Kaptelinin and Nardi (2006) combined some prior works into a book whose primary thesis involved characterising Vygotsky's work as individual-centric and distinguishing it from Leon'tiev's work, which they characterised as communal. Walls' 2009 review of the work found it to be an attack on the cognitive science approaches to HCI rather than containing any new contributions.

In attempting to relieve confusions often associated with AT, Uden et al. (2008) clarified that an activity may lose its 'motive' and thus be demoted (down through Leon'tev's hierarchy) to an action, once a problem or blockage is resolved. The actors' doings would become automated or 'frozen' into some form of regularity, which could be driven by a mutually understood goal. Whilst this important observation inferred that the 'demotion' of activities to actions may indicate the resolution of problems; for Uden however, it seems rather to

have been the reverse suggestion by Baerentsen and Trettvik (2002) that was exciting. Rather than *relieving* problem states down into mundane actions through resolution of blockages, Uden seemed particularly interested by the notion of *elevating* mundane workaday systems into 'living' and 'dynamic' structures able to *accommodate* such organically flexible and dynamic tasks. Her work involved informing design notions for flexible web application interfaces to permit richly dynamic navigational activities.

Since 2009 there has been an extraordinary paucity of academic articles in the SA&D space or the HCI space deploying , or even considering, AT as an informing methodological principle. The majority of more recent works deploy AT in it's more traditional role as an explanatory framework. By way of example, Tan and Melles reported in 2010 that AT was particularly useful for investigating the practices of graphical designers, and by drawing on the work of Nardi, they deployed an AT framework for analysing professional practices. Unfortunately, they offered no methodological commentary or contribution for the conduct of design.

Activity Theory

Approaches sensitive to the user viewpoint are valid as stakeholders generate their own notations and terminologies, complicating elicitation (Sommerville, Sawyer and Viller 1998). Significant risk of failure exists in marginalizing stakeholder's softer objectives, despite their inherent informality. If poor requirements are at least partially attributable to poor communications between phases, analysts, designers and users; then a case exists for a lightweight, readily learnable, methodologically flexible end-to-end approach, under a single theoretical framework (addressing user activity) that concentrates on the identification of requirements. Observing the shift of focus from technology to people under user-centric design, Constantine and Lockwood (1999) said "It is not users who must be understood, but usage." Räsänen and Nyce (2006) argued for anthropological analysis to avoid skewing the focus of analysis to individual users over their larger socio-structural processes. A roughly taxonomy of evolving centrisms might be given as: Product centric, Process centric, Goal centric, then User centric, which itself may be conceived as containing; Dependency centrism and Motive or Activity centrism. This final conception, termed use-centrism, conveys 'per user, engaged in work'.

AT identifies an activity as the smallest meaningful task carried out by a human subject. Vygotsky (1978) held that all human Activity is carried out by a Subject, using physical or psychological tools to achieve some object which may result in a physical outcome. Leont'ev (1978) proposed that all collective activities are directed to a single object (or motive). Within that abstract motive are more specific goal oriented actions. His three layer hierarchic structure of activity, action and operation, represent different levels of intellectual 'engagement'. At the base level, near-autonomic operations react to prevailing conditions. Leont'ev's notion describes doings abstracting to higher levels as the Subject devotes more cognitive attention upon them. Kuutti (1991) introduced a topmost abstraction: the activity network. Engström (1987) described the structure of each activity as a seven node matrix. Traditionally, AT is concerned with the cognitive ramifications of the differences between intended object and resultant outcome.

A number of Activities may reside near one another and interact, forming a network that describes a larger process (Kuutti 1991). The outcome of one activity may constitute (among other things) a tool in another (Vrazalic 2004). We are specifically interested in outcome-tool transactions. It is necessary to shift focus away from the psychosocial and cognitive aspects to the investigation of the facilitating tool(s) of an activity, as a subset of these could specify some new system.

Ultimately, the system may be specified by describing those outcome-tool transactions and transformations which may between activities. The designer must identify and describe them. These descriptions specify the requirements for functions to facilitate these transactions and transformations. We hope to isolate those which could usefully pass through some facilitating computer system, and we will use the term 'Instrument' to refer to data-artefacts that are passed in such transactions.

AT was not readily convertible into a workable Systems Analysis and Design (SA&D) method. Martins and Daltrini (1999) unhappily reporting that AT had not yet delivered any prescriptive methods. Otwell (2005) bemoaned a lack of prescriptive method for applying AT; saying "Activity Theory seems to almost defy practical application". Otwell specifically cited Collins' Activity Centered Design (ACD) work of 2002 as offering no concrete example of AT in use for design. Brown (2010) surveyed thirteen prior attempts to apply AT to some or all of the SA&D process and found that most were abandoned, converted into niche tools or, at best, delivered little more than a set of issues for systems designers to be mindful of in their work.

It must be acknowledged that although a system may not be what a client wants it could still be what they need. Such tensions could result in the 'failure' of an otherwise technically excellent product. There are grounds to suspect that SA&D methods to date have not bridged clients' inadequate grasp of IS and the analyst/designers' inadequate grasp of their client.

Taking the broadest brush abstraction of the Software Development Lifecycle (SDLC), we consider briefly the Analysis of what will be needed, the Design of a System to achieve that, and its Implementation. During any Analysis of 'what to make', there will be issues of translation between the analyst and the customers' paradigms. These continue as requirements are passed between the disciplines of Analysis and Design. Assuming that a viable Design can be arrived at, there will be issues of transcription between the Design and Implementation phases conveying the instructions 'make it like this'.

ATSA

Described at length elsewhere (Brown and Piper 2011) the Activity Theoretic System Architecture (ATSA) Method deconstructs users (members of some agenda-directed group, such as a business community) into their task-related activity "Roles" (it permits a many-to-many relationship between users and Roles). ATSA elicits the "doings" of each Role and identifies the data-like-things (Instruments) which they transact with each other in the course of their activities. ATSA then strives to facilitate this socio-technical complex (reducing the mental workload of mundane, repetitive tasks on users) by inserting one or more potential "systems" into the community. Many of the Instrument transactions may be more efficiently, securely or economically conducted via the agency of this "system actor". It its current form, a form of data dictionary emerges from an ATSA analysis which can specify the procedural coding of multi-user systems. Ongoing work indicates that an Object Oriented set of specifications may be easily generated as well.

Importantly, each time an Instrument crosses the system boundary (to or from a user) there must be some interface element. The system, as conceived under ATSA, is no more or less than the concatenation of these transactions and their interface elements. ATSA designs its systems from the outside-in, thus addressing one of Meyers' fundamental issues with HCI.

If the users are reorganised in some way, an ATSA conceived system simply reallocates functionality to new workstations, as the "doings" are still tied to Roles, which tend to remain invariant despite how they may be instantiated in actual people. To change the interface significantly is to change what the system does, and visa versa. A system specification under ATSA is, necessarily, also an HCI specification. The method exhibits considerable promise towards some reconciliation between currently disparate disciplines.

Though ATSA does not specify precisely what interface element must be deployed at any one transaction point (any more than it declares the code or language the system coder should deploy), it does present a clear notion of how interface elements should be clustered according to the context of use. Of itself, this is a significant result of considerable use. The most beautifully crafted interface elements are less useful when inappropriately grouped, forcing users to switch between screens or contexts (mental or literal).

Example of Interface Element Specification from an ATSA Analysis

A representative sample of data was drawn from a case study ATSA design. For the purposes of investigating ATSA's HCI value, it was decided to examine the position of Office Manager (OM) in closer detail as it was a middle management position with multiple, but specific authorised duties. At the time of analysis, this position performed eight identified roles, two of which are given here as Table 1 (below).

Under the ATSA method, the Job Administrator's (JobA's) activities were extracted as were instruments transacted by them. Two discrete activities (numbers 12 and 14) were partially assigned to the system. Activity 12 is where details of an upcoming job request are elicited and recorded, whilst in activity 14 a job sheet is activated and placed into a manual sequence of notice board postings. Instruments employed by the JobA are detailed in Table 2.

A possible selection and assembly of UI elements is suggested by the "re-designed" instrument transactions and the motivational constraints elicited in the earlier ATSA analysis. Figure 1, below, shows the primary interface for the Job Sheet used by the JobA, with some mock data for some sample job. On the left are all the primary data fields for the combined Job Sheet and Job Scope Report, and from here the JobA may perform data entry or modification as required. It is assumed that the Open Job Sheet button and the Save Job Sheet buttons would invoke standardized and familiar GUI combination widgets in the manner of a File Open and File SaveAs, and as such are not illustrated here. The combination file open widget however should incorporate the ability to sort through the list of jobs by: invoice number, client, start date, location, end date and especially by status (being one of Pending, Current or Ended).

The Print Job Sheet button would invoke a standard File Print dialogue and produce a hardcopy or PDF for emailing. The Job Map button invokes a pop-up window illustrated in Figure 2.

The Client Contact Details button would invoke a pop-up window displaying the contact information for the client associated with the job (with some possible future functionality to directly place a phone call via some

integrated data-voice business telecommunications network solution) illustrated in Figure 3. The Job Scope Questions button would invoke yet another pop-up question with a list of prompts to assist the JobA in eliciting required job details (Figure 4).

The screen has a configurable element whereby a number of employee placeholders is dynamically created according to the total number of needed employees selected in the widget in the upper right corner. The JobA is able to specify the required Qualification for each (according to information elicited from the client and/or directive given by the MD-Ops/CSO) and also nominate (using a radio button) which employee would be the supervisor for that job.

Each employee placeholder has a pull-down control (shown activated for the second employee) where any special comments or conditions pertinent to that particular placement may be recorded.

Some elements of this form are completed by other roles. The Chief Financial officer (CFO), for example, can complete the invoice number and select a charge rate, or simply instruct the JobA by offline means. The JobA can initiate an invoice number request with the call button beside the invoice number field. The client name, charge rate and job descriptions are all selectable (via a spin button) from pre-made lists created and maintained in other activities. The job start and end dates can be selected using a date-selection widget invoked with the small call-button beside those fields. The job status of PENDING, CURRENT or ENDED is automatically assigned by the system by comparing the start and end dates with the current system date 'TODAY'.

The JobA does not finalise this form completely, but prepares it (or rather the values of the various instruments it points to) for the employee selection process which is later conducted by the EmpA.

Table 1: Some roles currently assigned to the Office Manager

(JobA) Job Booking Administrator
May also be performed by the MD-Opns, or the AA (under supervision)
Takes enquiries from Clients and books the details of requested jobs.
(EmpA) Employee Placement Administrator
May also be performed by the MD-Opns, or the AA (under supervision)
Allocates employees to specific jobs, being careful to meet all operational and administrative requirements.

Table 2: JobA Instruments

Instrument	**Description**	**WHERE [Activity] {Role}**	
		Declared	**Used**
name	*Name*	[12] {JobA}	
description	*Description*	[12] {JobA}	
description	*job description*	[14] {JobA}	
location	*job Location*	[14] {JobA}	[17] {EmpA}
map	*job Map*	[14] {JobA}	
start	*job Start Date*	[14] {JobA}	
end	*job End Date*	[14] {JobA}	
status	*current status of particular job, according to current date and scheduled start/finish dates*	[AUTO]	[15] {JobA}
selection	*final selection determined?*	[..] {JobA}	[29] {JobA}
qualification	*Qual per person needed*	[12] {JobA}	[20] {EmpA}

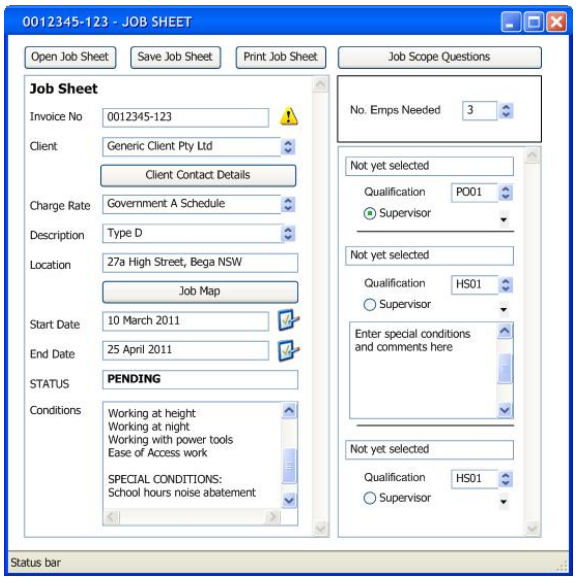
The interfaces given here are by no means claimed to be optimal, let alone elegant, but they are informed entirely by consideration of the activities analysed the instruments declared in the ATSA re-design phase.

The claim is that ATSA assembles sufficient data to permit a neophyte to assemble an adequate interface. The very act of analysing the client group's agenda and doings, then designing a set of data-like instruments, informs

a selection of interface elements, though an experienced UI designer might be expected to have a well developed stylistic palette to draw from.

Under AT-oriented elicitation and analysis it was found that the manner of the business and its agenda require that certain crucial sets of doings must be conducted under experienced human judgment, open to revision. Accordingly, the UI for the EmpA could allow controls for activities clustered, and might assume a degree of automation is possible, but must allow user intervention to over-ride automated many decisions and declare the final decision.

As for the JobA's UI, the Employee Administrator (EmpA) would require access to the Job Sheet for a specific job. Similarly, the EmpA will need to be able to search for a given job through a commonly understood combination File Open style widget, as described above.



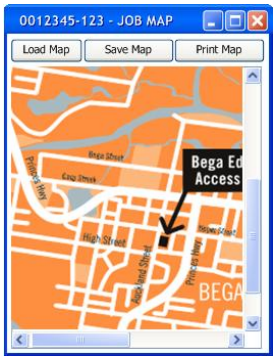


Figure 1: Job Sheet for sample job. Figure 2: Job Map for sample job

Figure 3: Client Contact Details for sample client
and Figure 4: Sample pop-up Job Scope Questions list

Upon opening a specific job, the EmpA will be presented with a listing of all employees in a sortable list. The system will have pre-sorted them by their status colour (green, amber or red) and will have, by default, unselected those with a red status. To reduce screen clutter, the UI as imagined by this author, red-status

employees will simply not be displayed, though a button will allow them to be displayed if required (remembering that the CSO has the authority to reclassify employees at any time, and/or to deploy a red-status employee under the supervision of another more trusted employee). Should otherwise unusable employees be displayed, small warning icons will be displayed beside their names.

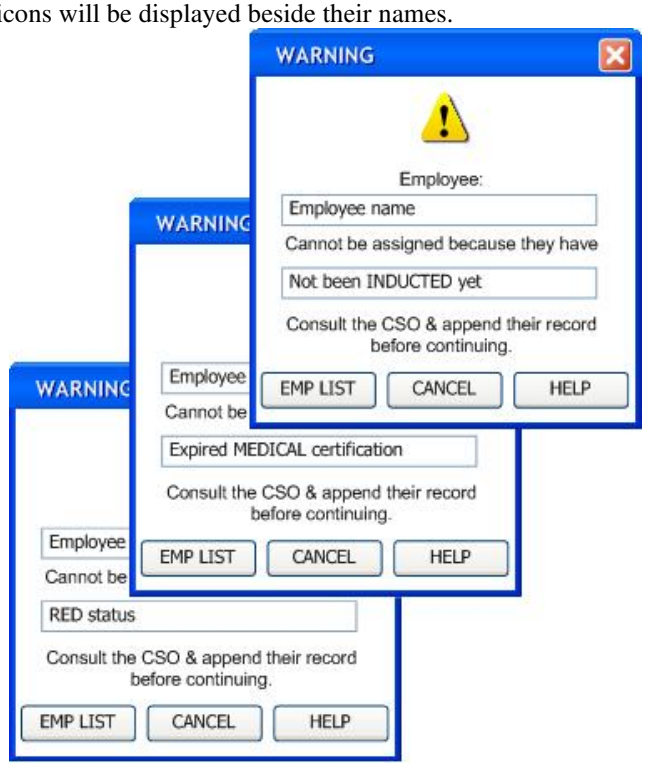


Figure 5: Confirmation Dialogues to prevent illegal assignments

Similar mechanisms allow for the system to automatically hide listings of employees who lack current medical clearance or induction status, though they may be displayed on request as in exceptional circumstances, the CSO may decide to allow them time to become compliant with these pre-conditions if time permits before the job start date. Each of these categories of, otherwise, unusable employees must be flagged for attention to avoid accidental assignment of legally unusable employees.

Assignment is visualised as a click-and-drag from the list in the Employee Selection Tool to the appropriate slot in the Job Sheet window. Any assignment will trigger a confirmation dialogue but the system will automatically display an appropriate warning whenever any unsuitable employee is assigned to a job, highlighting critical issues such as red status, expired medical clearance or a lack of induction. These are illustrated below in Figure 5. The dialogue allows the employee record to be amended (under the authority of the CSO) which would then permit assignment. The master Employee List will record the date and time of all amendments to track such alterations.

The Employee Selection Tool (given in Figure 6) will present action buttons which open pop-up windows providing further information. By simply highlighting an employee and clicking, the EmpA may view the employee file (with their contact details), the employee's job history (listing the jobs they've been assigned to previously) and their entry in the job experience register.

The tool will sort first by colour status, putting all green level employees at the top, then amber (with red hidden by default). Within that, they will be sorted alphabetically by last name, and then by the distance between their default location (home address) and the job location. This distance will be calculated automatically by a series of automated GoogleMap enquiries conducted by the system using some form of web-service architecture. Should it prove too expensive or too burdensome on network or processor resources, the system may be modified to only calculate these distances upon request. This is a technical issue of implementation, beyond the direcrt scope of this study. The EmpA may also simply highlight the header of any field and use the Sort button to re-order the listing, as required.

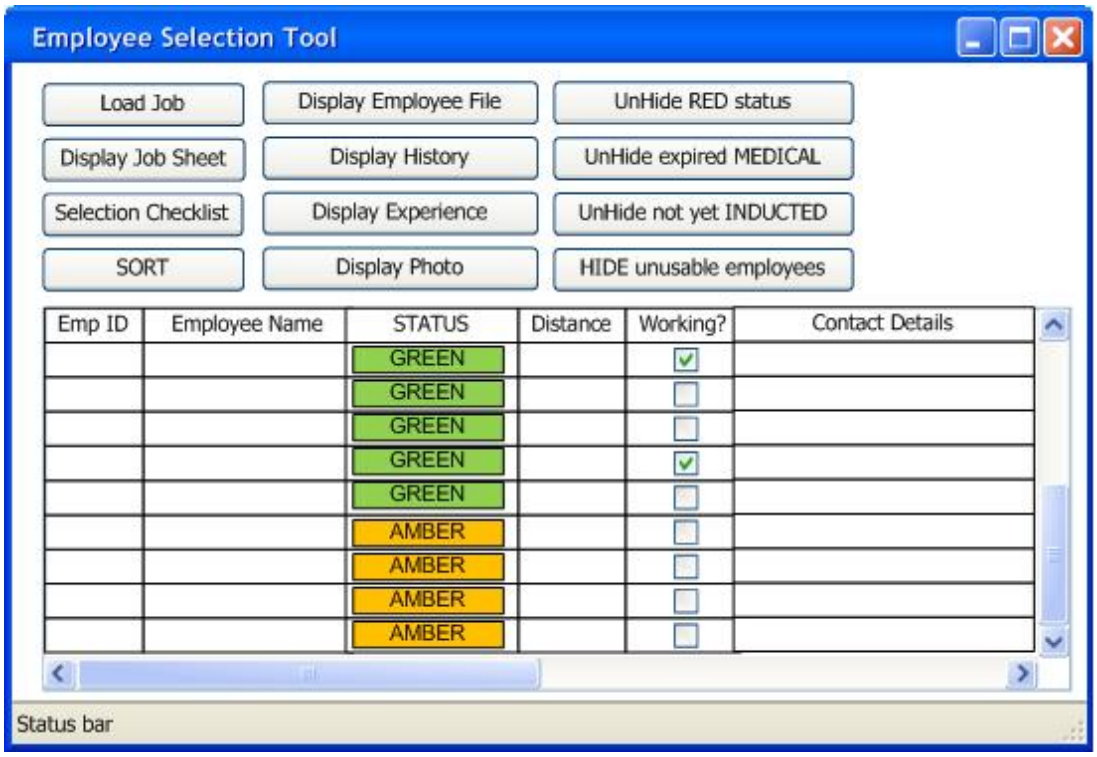


Figure 6: Employee Selection Tool

CONCLUSION

The ATSA method suggests a possible vector towards reconciling HCI and the broader SA&D disciplines. By analysing business groups according to their members' communal Role-based doings and the data-like transactions they conduct, it offers a specification of both a system and its interface in a uniquely integrated manner.

By recognising the relative invariance of "doings" (rather than the identity or configuration of the do-ers themselves), ATSA analysis offers an opportunity to approach the design of interface elements with a clear knowledge of context and use-driven groupings.

To users, information systems largely "are" their interfaces and the experience of using them, so the ATSA Method is suggested as one possible vector towards instantiating this insight in the practice of information system analysis and design.

REFERENCES

Baerentsen, K.B. and Trettvik, J. 2002. "An activity theory approach to affordance" *Proceedings of the second Nordic conference on Human-computer interaction table of contents Aarhus, Denmark.* pp. 51-60. New York, NY: ACM Press.

Bannon, L. J. and Bødker, S. 1991. "Beyond the Interface: Encountering Artifacts in Use", *in* Carroll, J. (ed.) *Designing Interaction: Psychology at the human-computer interface*, New York: Cambridge U.P., Chapter 12.

Bardram, J. 1998. "Designing for the dynamics of cooperative work activities" *Proceedings of the 1998 ACM conference on Computer Supported Cooperative Work*, pp 89-98.

Benyon, D., Turner, P. and Turner, S. 2005. *Designing Interactive Systems: people Activities, Contexts, Technologies*, Addison-Wesley, London, UK.

Beyer, H., and Holtzblatt, K. 1998. *Contextual design: Defining customer-centered systems*. San Francisco: Morgan Kaufmann

Bødker, S. 1991. *Through the Interface: A human activity approach to interface design*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Brown R.B.K. 2010, *the ATSA Method for Specifying both System and User Interface Requirements: An Application of Activity Theory*, PhD Thesis, University of Wollongong, Australia.

Brown R.B.K., Piper I.C. 2011, "What users Do: SA&D with the ATSA Method", presented at the *20th International Conference on Information Systems Development (ISD2011),* Edinburgh UK (publication pending).

Dix, A., Finlay, J., Abowd, G. and Beale, R. 2004. *Human Computer Interaction* (3rd ed). Prentice Hall, Harlow, UK.

Draper, S. and Norman, D. 1984. "Software Engineering for User Interfaces" *Proceedings of the 7th International Conference on Software Engineering*, Silver Spring, MD:IEEE Press, pp. 214-220.

Engström, Y. 2001. "Expansive learning at work: towards an activity theoretical reconceptualization" *Journal of Education and Work,* **14**(1), pp.133-156.

Grudin, J. 1989. "The case against user interface consistency" *Communications of the. ACM*, 32(10), pp,1164–1173.

Kaptelinin, V. 2005, "The Object of Activity: Making Sense of the Sense-Maker", *Mind, Culture, and Activity*, **12**(1), pp.4-18.

Kaptelinin, V., Nardi, B. A. 2009. *Acting with technology : activity theory and interaction design.*MIT Press, Cambridge, Mass.

Kuutti K. 1991. "Activity Theory and its applications to information systems research and development" Nissen HE, Klein HK, and Hirsheims R (eds) *Information Systems Research: Contemporary Approaches and Emergent Traditions*, Elsevier Science, Amsterdam, pp.529-549.

Lauesen, S. 2005. *User Interface Design: A Software Engineering Perspective*, Addison Wesley,London, UK.

Leont'ev, A.N. 1978. *Activity, Consciousness, and Personality*, Prentice Hall.

Linberg, K. R. 1999. "Software developer perceptions about software project failure: a case study", *Journal of Systems and Software*, **49**(2-3), pp. 177-192.

Martins L.E.G. and Daltrini B.M. 1999. "An Approach to Software Requirements Elicitation Using Precepts from Activity Theory". *4th IEEE International Conference on Automated Software Engineering*, pp.15-23.

Myers, B. A. 1993. "Why are Human-Computer Interfaces Difficult to Design and Implement?" *Technical Report CS-93-183*, School of Computer Science, Carnegie Mellon University.

Nardi, B. A. 200) "Objects of Desire: Power and Passion in Collaborative Activity" *Mind, Culture, and Activity*, **12**(1), pp.37-51.

Nielsen, J. 1994. *Usability Engineering*, Morgan Kaufmann, San Francisco.

Olsen, D. R. Jnr. 1998. *Developing User Interfaces*, Morgan Kaufmann, San Francisco, CA.

Otwell A. 2005, *Activity Theory and User-centered Design. Blog: heyblog: A Space for half-formed Thoughts.* (July 29, 2005). http://www.heyotwell.com/heyblog/archives/2005/07/ activity_theory.html [accessed 20th August 2008]

Platt, D. S. 2007. *Why Software Sucks … and what you can do about it*, Addison-Wesley.

Preece, J., Rogers, Y. and Sharp, H. 2002, *Interaction Design: beyond human-computer interaction*, John Wiley & Sons,Inc., New York, NY.

Räsänen M. and Nyce J.M. 2006, "A New Role for Anthropology? – rewriting 'Context' and 'Analysis' in HCI" *Nordic Conference on Human-Computer Interaction: Changing Roles (NORDICHI'06)*, Oslo, Norway, Proceedings New York, NY, USA. ACM Press pp. 175-184.

Rosson, M. B. and Carroll, J. M. 2002 *Usability Engineering: scenario-based development of human computer interaction*, Morgan Kaufmann, Redwood City, CA.

Shelly, G. B. and Rosenblatt, H. J. 2011 (8th ed. advanced copy), *Systems Analysis and Design*, Cengage Learning, Boston, MA.

Shneiderman, B. and Plaisant, C. 2005, *Designing the User Interface*, (4th ed), Pearson Addison Wesley, Boston, MA.

Sommerville I., Sawyer P., and Viller S. 1998. "Viewpoints for requirements elicitation: a practical approach" *Proceedings of the 3rd IEEE international conference on requirements engineering (ICRE'98)*, Colorado Springs, USA, pp.74 – 81.

Tan S. and Melles G. (2010) "An activity theory focused case study of graphic designers' tool-mediated activities during the conceptual design phase", *Design Studies, **31**(*5) pp. 461-478

Thimbleby, H. 1990, *User Interface Design*, ACM Press, New York, NY.

Torres, R. J. 2002, *Practitione'rs Handbook ForUser Interface Design and Development*, Prentice Hall, Upper Saddle River, NJ.

Uden, L., Valderas, P. and Pastor, O. 2008. "An activity-theory-based model to analyse Web application requirements" *Information Research*, **13**(2)

Verenikina I and Gould E 1998. "Cultural-Historical Psychology and Activity Theory" In: Hasan H, Gould E and Hyland P (eds.) *Information Systems and Activity Theory*, 1st edn. University of Wollongong Press.

Vrazalic, L. 2004. *Towards Holistic Human-Computer Interaction Evaluation Research and Practice: Development and Validation of the Distributed Usability Evaluation Method.* PhD Thesis, University of Wollongong, Australia.

Walls, D. M. 2009. Book Review: Kaptelinin, Victor, and Nardi, Bonnie A. 2006 "Acting With Technology: Activity Theory and Interaction Design", *Journal of Business and Technical Communication* **23**(380).

Wilson, T.D. 2006. Review of: Kaptelinin, V. & Nardi, B.A. *Acting with technology: activity theory and interaction design* Cambridge, MA: MIT Press, in *Information Research,* **12**(2), review no. R247.

Wright, P., Merriam, N. and Fields, B. 1997. "From formal models to empirical evaluation and back again" *Formal Methods in HCI*, Paterno and Palanque (eds). Springer.

COPYRIGHT